# Prime 1.5

User Manual

Schrödinger Press

Revision A, April 2006

# Contents

# Document Conventions

In addition to the use of italics for names of documents, the font conventions that are used in this document are summarized in the table below.

| Font | Example | Use |
|---|---|---|
| Sans serif | Project Table | Names of GUI features, such as panels, menus, menu items, buttons, and labels |
| Monospace | `$SCHRODINGER/maestro` | File names, directory names, commands, environment variables, and screen output |
| Italic | *filename* | Text that the user must replace with a value |
| Sans serif uppercase | CTRL+H | Keyboard keys |

In descriptions of command syntax, the following UNIX conventions are used: braces `{ }` enclose a choice of required items, square brackets `[ ]` enclose optional items, and the bar symbol `|` separates items in a list from which one item must be chosen. Lines of command syntax that wrap should be interpreted as a single command.

In this document, to *type* text means to type the required text in the specified location, and to *enter* text means to type the required text, then press the ENTER key.

References to literature sources are given in square brackets, like this: [10].

# Introduction

Prime 1.5 is a highly accurate protein structure prediction suite of programs that integrates Comparative Modeling and Threading into a single user-friendly, wizard-like interface. The Comparative Modeling path incorporates the complete protein structure prediction process from template identification, to alignment, to model building, and finally to refinement. Refinement involves side-chain prediction, loop prediction, and minimization. The Threading path takes a sequence through a Fold Recognition module to alignment, model building, and refinement.

The Prime interface was designed to accommodate both novice and expert users, while the underlying programs were designed to produce superior results in a variety of applications, including:

- High-resolution homology modeling
- Refinement of active sites
- Induced-fit optimization
- Threading/Fold Recognition
- Structure-based functional annotation
- Generation of alternate loop conformations

## 1.1 The Prime Suite in Maestro

To run Prime you use Maestro, the graphical user interface (GUI) for all of Schrödinger's products. For an introduction to Maestro, see Chapter 2. For help with a Maestro panel, click the Help button. For more information, see the *Maestro User Manual*.

The Prime submenu of the Maestro Applications menu has two items, Structure Prediction and Refinement, which open the Prime–Structure Prediction (Prime–SP) and Prime–Refinement modules of the Prime suite.

Prime–SP consists of a branched series of steps leading from a protein sequence through Comparative Modeling or Threading to the construction and refinement of a 3D structure. If a template of sufficiently high sequence identity exists in the database of known structures, the Comparative Modeling path is used. If a suitable template cannot be identified using sequence information, however, the Threading path can be used, in which Fold Recognition is used to identify potential templates. As step follows step, Prime–SP displays the appropriate step panels.

Prime–Refinement is a stand-alone facility for refining protein structures that have been imported into Maestro or added to the Project Table from a Schrödinger application. Although it offers the same refinement protocols available in the Refine Structure step in Prime–SP, it is run from its own panel. For more about Prime–Refinement, see Chapter 11.

## 1.2 The *Prime User Manual*

The *Prime User Manual* describes the Prime suite, including information about:

- Opening the Prime–SP and Prime–Refinement modules
- Becoming familiar with the Maestro GUI
- Essentials of using the Prime–SP module: runs and projects, job control, and the layout and common features of step panels
- Starting a structure prediction: the Input Sequence and Find Homologs steps
- Using the Comparative Modeling path, including the Edit Alignment, Build Structure, and Refine Structure steps
- Using the Threading path, including the Fold Recognition, Build Backbone, and Refine Backbone steps
- Using the stand-alone Prime–Refinement module

This document expands on the information provided in the Maestro online help (see Section 1.3). In addition to describing steps, features, and options, this document suggests ways to determine which choices are the most useful, describes results, and provides some background and technical detail. Terms pertinent to Prime are defined in the Glossary.

Other manuals that may be useful include the *Prime Quick Start Guide*, the *Installation Guide*, and the *Maestro User Manual*.

For information about Schrödinger's Induced Fit Docking protocol, which uses Prime and Glide to optimize docking by including ligand-induced flexibility in the receptor active site, see the document *Induced Fit Docking*.

## 1.3 Online Help

Maestro online help is available for Prime.

**To view the Help topic for the Prime–SP step in which you are working:**

- Click the Help button in the lower right portion of the panel.

**To view a list of all Prime–related topics in the Help system:**

1. Choose Help from the Help menu on the Maestro menu bar to open the Help panel.
2. Click the Categories tab.
3. From the Categories menu, choose Prime.

   Topics are listed, including a topic for each step panel in Prime–SP, general usage information topics, and a topic for the Prime–Refinement module.
4. Double-click the title of the topic you want to view.

Each Prime Help topic has links to related topics in Prime and in Maestro. See Section 2.11 on page 30 for a discussion of help in Maestro, including tooltips (Balloon Help) and technical support contacts.

## 1.4 Citing Prime in Publications

The use of this product should be acknowledged in publications as:

Prime, version 1.5, Schrödinger, LLC, New York, NY, 2005.

# Introduction to Maestro

Maestro is the graphical user interface for all of Schrödinger's products: CombiGlide™, Epik™, Glide™, Impact™, Jaguar™, Liaison™, LigPrep™, MacroModel®, Phase™, Prime™, QikProp™, QSite™, and Strike™. It contains tools for building, displaying, and manipulating chemical structures; for organizing, loading, and storing these structures and associated data; and for setting up, monitoring, and visualizing the results of calculations on these structures. This chapter provides a brief introduction to Maestro and some of its capabilities. For more information on any of the topics in this chapter, see the *Maestro User Manual*.

## 2.1 General Interface Behavior

Most Maestro panels are amodal: more than one panel can be open at a time, and a panel need not be closed for an action to be carried out. Each Maestro panel has a Close button so you can hide the panel from view.

Maestro supports the mouse functions common to many graphical user interfaces. The left button is used for choosing menu items, clicking buttons, and selecting objects by clicking or dragging. This button is also used for resizing and moving panels. The right button displays a shortcut menu. Other common mouse functions are supported, such as using the mouse in combination with the SHIFT or CTRL keys to select a range of items and select or deselect a single item without affecting other items.

In addition, the mouse buttons are used for special functions described later in this chapter. These functions assume that you have a three-button mouse. If you have a two-button mouse, ensure that it is configured for three-button mouse simulation (the middle mouse button is simulated by pressing or holding down both buttons simultaneously).

## 2.2 Starting Maestro

Before starting Maestro, you must first set the `SCHRODINGER` environment variable to point to the installation directory. To set this variable, enter the following command at a shell prompt:

| | |
|---|---|
| **csh/tcsh:** | `setenv SCHRODINGER` *installation-directory* |
| **bash/ksh:** | `export SCHRODINGER=`*installation-directory* |

You might also need to set the DISPLAY environment variable, if it is not set automatically when you log in. To determine if you need to set this variable, enter the command:

```
echo $DISPLAY
```

If the response is a blank line, set the variable by entering the following command:

| | |
|---|---|
| **csh/tcsh:** | `setenv DISPLAY` *display-machine-name*`:0.0` |
| **bash/ksh:** | `export DISPLAY=`*display-machine-name*`:0.0` |

After you set the SCHRODINGER and DISPLAY environment variables, you can start Maestro using the command:

```
$SCHRODINGER/maestro options
```

If you add the $SCHRODINGER directory to your path, you only need to enter the command maestro. Options for this command are given in Section 2.1 of the *Maestro User Manual*.

The directory from which you started Maestro is Maestro's current working directory, and all data files are written to and read from this directory unless otherwise specified (see Section 2.8 on page 27). You can change directories by entering the following command in the command input area (see page 8) of the main window:

```
cd directory-name
```

where *directory-name* is either a full path or a relative path.

## 2.3 The Maestro Main Window

The Maestro main window is shown in Figure 2.1 on page 7. The main window components are listed below.

The following components are always visible:

- **Title bar**—displays the Maestro version, the project name (if there is one) and the current working directory.
- **Auto-Help**—automatically displays context-sensitive help.
- **Menu bar**—provides access to panels.
- **Workspace**—displays molecular structures and other 3D graphical objects.

The following components can be displayed or hidden by choosing the component from the Display menu. Your choice of which main window components are displayed is persistent between Maestro sessions.

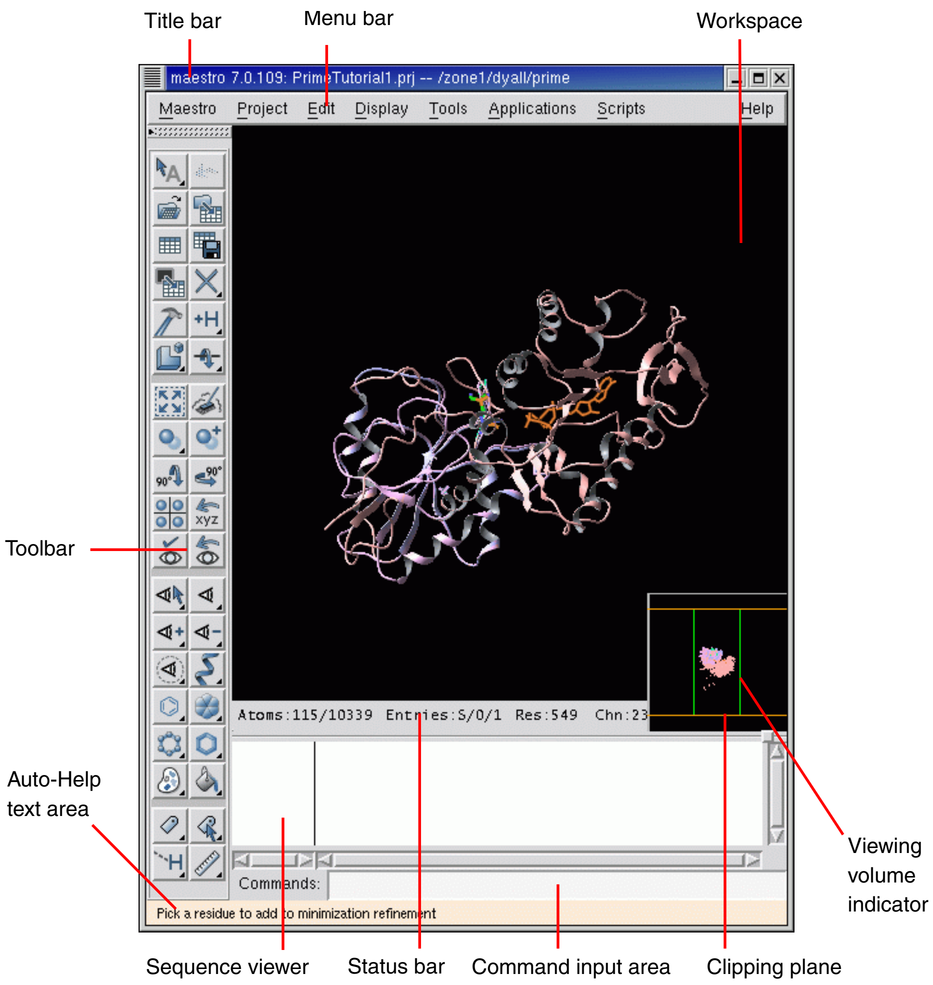


***Figure 2.1. The Maestro main window.***

- **Toolbar**—contains buttons for many common tasks and provides tools for displaying and manipulating structures, as well as organizing the Workspace.
- **Status bar**—displays information about a particular atom, or about structures in the Workspace, depending on where the pointer pauses (see Section 2.5 of the *Maestro User Manual* for details):
  - **Atom**—displays the chain, residue number, element, PDB atom name, formal charge, and title or entry name (this last field is set by choosing Preferences from the Maestro menu and selecting the Feedback folder).

    - **Workspace**—displays the number of atoms, entries, residues, chains, and molecules in the Workspace.

- **Clipping planes window**—displays a small, top view of the Workspace and shows the clipping planes and viewing volume indicators.
- **Sequence viewer**—shows the sequences for proteins displayed in the Workspace. See Section 2.6 of the *Maestro User Manual* for details.
- **Command input area**—provides a place to enter Maestro commands.

When a distinction between components in the main window and those in other panels is needed, the term *main* is applied to the main window components (e.g., main toolbar).

You can expand the Workspace to occupy the full screen, by pressing CTRL+=. All other components and panels are hidden. To return to the previous display, press CTRL+= again.

## 2.3.1 The Menu Bar

The menus on the main menu bar provide access to panels, allow you to execute commands, and control the appearance of the Workspace. The main menus are as follows:

- Maestro—save or print images in the Workspace, execute system commands, save or load a panel layout, set preferences, set up Maestro command aliases, and quit Maestro.
- Project—open and close projects, import and export structures, make a snapshot, and annotate a project. These actions can also be performed from the Project Table panel. For more information, see Section 2.4 on page 13.
- Edit—undo actions, build and modify structures, define command scripts and macros, and find atoms in the Workspace.
- Display—control the display of the contents of the Workspace, arrange panels, and display or hide main window components.
- Tools—group atoms; measure, align, and superimpose structures; and view and visualize data.
- Applications—set up, submit, and monitor jobs for Schrödinger's computational programs. Some products have a submenu from which you can choose the task to be performed.
- Scripts—manage and install Python scripts that come with the distribution and scripts that you create yourself. (See Chapter 13 of the *Maestro User Manual* for details.)
- Help—open the Help panel, the PDF documentation index, or information panels; run a demonstration; and display or hide Balloon Help (tooltips).

### 2.3.2 The Toolbar

The main toolbar contains three kinds of buttons for performing common tasks:

**Action**—Perform a simple task, like clearing the Workspace.

**Display**—Open or close a panel or open a dialog box, such as the Project Table panel.

**Menu**—Display a *button menu*. These buttons have a triangle in the lower right corner.

There are four types of items on button menus, and all four types can be on the same menu (see Figure 2.2):

- **Action**—Perform an action immediately.
- **Display**—Open a panel or dialog box.
- **Object types for selection**—Choose Atoms, Bonds, Residues, Chains, Molecules, or Entries, then click on an atom in the Workspace to perform the action on all the atoms in that structural unit.

  The object type is marked on the menu with a red diamond and the button is indented to indicate the action to be performed.
- **Other setting**—Set a state, choose an attribute, or choose a parameter and click on atoms in the Workspace to display or change that parameter.

The toolbar buttons are described below. Some descriptions refer to features not described in this chapter. See the *Maestro User Manual* for a fuller description of these features.

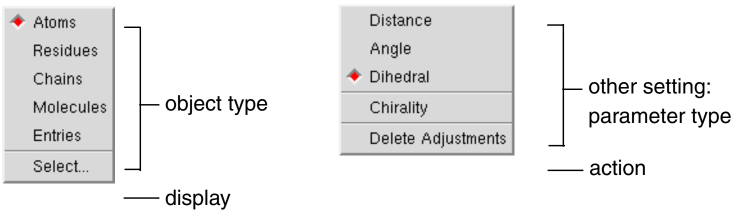


***Figure 2.2. The*** Workspace selection ***button menu and the*** Adjust distances, angles or dihedrals ***button menu.***

Workspace selection
– Choose an object type for selecting
– Open the Atom Selection dialog box

Undo/Redo
Undo or redo the last action. Performs the same function as the Undo item on the Edit menu, and changes to an arrow pointing in the opposite direction when an Undo has been performed, indicating that its next action is Redo.

Open a project
Open the Open Project dialog box.

Import structures
Open the Import panel.

Open/Close Project Table
Open the Project Table panel or close it if it is open.

Save as
Open the Save Project As dialog box, to save the project with a new name.

Create entry from Workspace
Open a dialog box in which you can create an entry in the current project using the contents of the Workspace.

Delete
– Choose an object type for deletion
– Delete hydrogens and waters
– Open the Atom Selection dialog box
– Delete other items associated with the structures in the Workspace
– Click to select atoms to delete
– Double-click to delete all atoms

Open/Close Build panel
Open the Build panel or close it if it is open.

Add hydrogens
– Choose an object type for applying a hydrogen treatment
– Open the Atom Selection dialog box
– Click to select atoms to treat
– Double-click to apply to all atoms

Local transformation
– Choose an object type for transforming
– Click to select atoms to transform
– Open the Advanced Transformations panel

Adjust distances, angles or dihedrals
– Choose a parameter for adjusting
– Delete adjustments

Fit to screen
Scale the displayed structure to fit into the Workspace and reset the center of rotation.

Clear Workspace
Clear all atoms from the Workspace.

Set fog display state
Choose a fog state. Automatic means fog is on when there are more than 40 atoms in the Workspace, otherwise it is off.

Enhance depth cues
Optimize fogging and other depth cues based on what is in the Workspace.

Rotate around X axis by 90 degrees
Rotate the Workspace contents around the X axis by 90 degrees.

Rotate around Y axis by 90 degrees
Rotate the Workspace contents around the Y axis by 90 degrees.

Tile entries
Arrange entries in a rectangular grid in the Workspace.

Reset Workspace
Reset the rotation, translation, and zoom of the Workspace to the default state.

Save view
Save the current view of the Workspace: orientation, location, and zoom.

Restore view
Restore the last saved view of the Workspace: orientation, location, and zoom.

Display only selected atoms
– Choose an object type for displaying
– Click to select atoms to display
– Double-click to display all atoms

Display only
– Choose a predefined atom category
– Open the Atom Selection dialog box

Also display
– Choose a predefined atom category
– Open the Atom Selection dialog box

Undisplay
– Choose a predefined atom category
– Open the Atom Selection dialog box

Display residues within N angstroms of currently displayed atoms
– Choose a radius
– Open a dialog box to set a value

Show, hide, or color ribbons
– Choose to show or hide ribbons
– Choose a color scheme for coloring ribbons

Draw bonds in wire
– Choose an object type for drawing bonds in wire representation
– Open the Atom Selection dialog box
– Click to select atoms for representation
– Double-click to apply to all atoms

Draw atoms in CPK
– Choose an object type for drawing bonds in CPK representation
– Open the Atom Selection dialog box
– Click to select atoms for representation
– Double-click to apply to all atoms

Draw atoms in Ball & Stick
– Choose an object type for drawing bonds in Ball & Stick representation
– Open the Atom Selection dialog box
– Click to select atoms for representation
– Double-click to apply to all atoms

Draw bonds in tube
– Choose an object type for drawing bonds in tube representation
– Open the Atom Selection dialog box
– Click to select atoms for representation
– Double-click to apply to all atoms

Color all atoms by scheme
Choose a predefined color scheme.

Color residue by constant color
– Choose a color for applying to residues
– Click to select residues to color
– Double-click to color all atoms

Label atoms
– Choose a predefined label type
– Delete labels

Label picked atoms
– Choose an object type for labeling atoms
– Open the Atom Selection dialog box
– Open the Atom Labels panel at the Composition folder
– Delete labels
– Click to select atoms to label
– Double-click to label all atoms

Display H-bonds
– Choose bond type:
intra—displays H-bonds within the selected molecule
inter—displays H-bonds between the selected molecule and all other atoms.
– Delete H-bonds
– Click to select molecule

Measure distances, angles or dihedrals
– Choose a parameter for displaying measurements
– Delete measurements
– Click to select atoms for measurement

### 2.3.3 Mouse Functions in the Workspace

The left mouse button is used for selecting objects. You can either click on a single atom or bond, or you can drag to select multiple objects. The right mouse button opens shortcut menus, which are described in Section 2.7 of the *Maestro User Manual*.

The middle and right mouse buttons can be used on their own and in combination with the SHIFT and CTRL keys to perform common operations, such as rotating, translating, centering, adjusting, and zooming.

*Table 2.1. Mapping of Workspace operations to mouse actions.*

| Mouse Button | Keyboard | Motion | Action |
|---|---|---|---|
| Left | | click, drag | Select |
| Left | SHIFT | click, drag | Toggle the selection |
| Middle | | drag | Rotate about X and Y axes<br>Adjust bond, angle, or dihedral |
| Middle | SHIFT | drag vertically | Rotate about X axis |
| Middle | SHIFT | drag horizontally | Rotate about Y axis |
| Middle | CTRL | drag horizontally | Rotate about Z axis |
| Middle | SHIFT + CTRL | drag horizontally | Zoom |
| Right | | click | Spot-center on selection |
| Right | | click and hold | Display shortcut menu |
| Right | | drag | Translate in the X-Y plane |
| Right | SHIFT | drag vertically | Translate along the X axis |
| Right | SHIFT | drag horizontally | Translate along the Y axis |
| Right | CTRL | drag horizontally | Translate along the Z axis |
| Middle & Right | | drag horizontally | Zoom |

### 2.3.4 Shortcut Key Combinations

Some frequently used operations have been assigned shortcut key combinations. The shortcuts available in the main window are described in Table 2.2.

*Table 2.2. Shortcut keys in the Maestro main window.*

| Keys | Action | Equivalent Menu Choices |
|---|---|---|
| CTRL+B | Open Build panel | Edit > Build |
| CTRL+C | Create entry | Project > Create Entry From Workspace |
| CTRL+E | Open Command Script Editor panel | Edit > Command Script Editor |
| CTRL+F | Open Find Atoms panel | Edit > Find |
| CTRL+H | Open Help panel | Help > Help |
| CTRL+I | Open Import panel | Project > Import Structures |
| CTRL+M | Open Measurements panel | Tools > Measurements |
| CTRL+N | Create new project | Project > New |
| CTRL+O | Open project | Project > Open |
| CTRL+P | Print | Maestro > Print |
| CTRL+Q | Quit | Maestro > Quit |
| CTRL+S | Open Sets panel | Tools > Sets |
| CTRL+T | Open Project Table panel | Project > Show Table |
| CTRL+W | Close project | Project > Close |
| CTRL+Z | Undo/Redo last command | Edit > Undo/Redo |
| CTRL+= | Enter and exit full screen mode (Workspace occupies full screen) | None |

## 2.4 Maestro Projects

All the work you do in Maestro is done within a *project*. A project consists of a set of *entries*, each of which contains one or more chemical structures and their associated data. In any Maestro session, there can be only one Maestro project open. If you do not specify a project when you start Maestro, a *scratch* project is created. You can work in a scratch project without saving it, but you must save it in order to use it in future sessions. When you save or close a project, all the view transformations (rotation, translation, and zoom) are saved with it. When you close a project, a new scratch project is automatically created.

Likewise, if there is no entry displayed in the Workspace, Maestro creates a *scratch* entry. Structures that you build in the Workspace constitute a scratch entry until you save the structures as project entries. The scratch entry is not saved with the project unless you explicitly add it to the project. However, you can use a scratch entry as input for some calculations.

To add a scratch entry to a project, do one of the following:

- Click the Create entry from Workspace button:

- Choose Create Entry from Workspace from the Project menu.
- Press CTRL+C.

In the dialog box, enter a name and a title for the entry. The entry name is used internally to identify the entry and can be modified by Maestro. The title can be set or changed by the user, but is not otherwise modified by Maestro.

Once an entry has been incorporated into the project, its structures and their data are represented by a row in the Project Table. Each row contains the row number, an icon indicating whether the entry is displayed in the Workspace (the In column), the entry title, a button to open the Surfaces panel if the entry has surfaces, the entry name, and any entry properties. The row number is not a property of the entry.

Entries can be collected into groups, and the members of the group can be displayed or hidden. Most additions of multiple entries to the Project Table are done as entry groups.

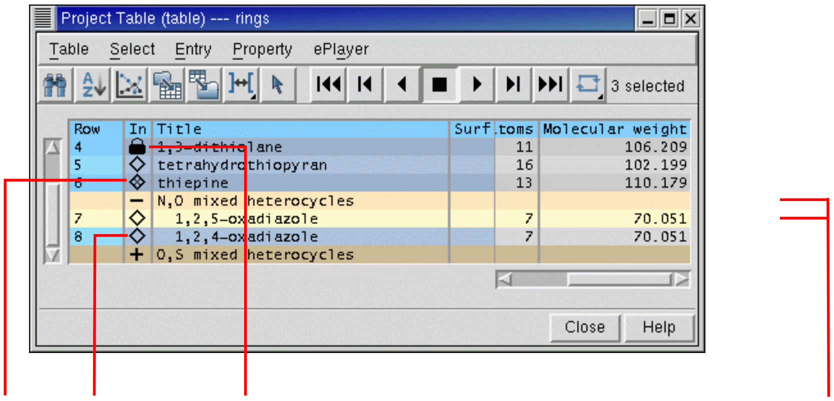
You can use entries as input for all of the computational programs—Glide, Impact, Jaguar, Liaison, LigPrep, MacroModel, Phase, Prime, QikProp, QSite, and Strike. You can select entries as input for the ePlayer, which displays the selected structures in sequence. You can also duplicate, combine, rename, and sort entries; create properties; import structures as entries; and export structures and properties from entries in various formats.

To open the Project Table panel, do one of the following:

- Click the Open/Close Project Table button on the toolbar

- Choose Show Table from the Project menu
- Press CTRL+T.

The Project Table panel contains a menu bar, a toolbar, and the table itself.



| Row | In | Title | Surf | .toms | Molecular weight |
|---|---|---|---|---|---|
| 4 | 🔒 | 1,3-dithiolane | | 11 | 106.209 |
| 5 | ◇ | tetrahydrothiopyran | | 16 | 102.199 |
| 6 | ◈ | thiepine | | 13 | 110.179 |
| | − | N,O mixed heterocycles | | | |
| 7 | ◇ | 1,2,5-oxadiazole | | 7 | 70.051 |
| 8 | ◇ | 1,2,4-oxadiazole | | 7 | 70.051 |
| | + | O,S mixed heterocycles | | | |

***Figure 2.3. The* Project Table *panel.***

### 2.4.1 The Project Table Toolbar

The Project Table toolbar contains two groups of buttons and a status display. The first set of buttons opens various panels that allow you to perform functions on the entries in the Project Table. The second set of buttons controls the ePlayer, which “plays through” the selected structures: each structure is displayed in the Workspace in sequence, at a given time interval. See Section 2.3.2 on page 9 for a description of the types of toolbar buttons. The buttons are described below.

Find
Open the Find panel for locating alphanumeric text in any column of the Project Table, except for the row number.

Sort
Open the Sort panel for sorting entries by up to three properties.

Plot
Open the Plot panel for plotting entry properties.

Import Structure
Open the Import panel for importing structures into the project.

Export Structure
Open the Export panel for exporting structures to a file.

Columns
Choose an option for adjusting the column widths.

Select only
Open the Entry Selection dialog box for selecting entries based on criteria for entry properties.

Go to start
Display the first selected structure.

Previous
Display the previous structure in the list of selected structures.

Play backward
Display the selected structures in sequence, moving toward the first.

Stop
Stop the ePlayer.

Play forward
Display the selected structures in sequence, moving toward the last.

Next
Display the next structure in the list of selected structures.

Go to end
Display the last selected structure.

Loop
Choose an option for repeating the display of the structures. Single Direction displays structures in a single direction, then repeats. Oscillate reverses direction each time the beginning or end of the list is reached.

The status display, to the right of the toolbar buttons, shows the number of selected entries. When you pause the cursor over the status display, the Balloon Help shows the total number of entries, the number shown in the table, the number selected, and the number included in the Workspace.

## 2.4.2 The Project Table Menus

- Table—find text, sort entries, plot properties, import and export structures, and configure the Project Table.
- Select—select all entries, none, invert your selection, or select classes of entries using the Entry Selection dialog box and the Filter panel.

- Entry—include or exclude entries from the Workspace, display or hide entries in the Project Table, and perform various operations on the selected entries.
- Property—display and manipulate entry properties in the Project Table.
- ePlayer—view entries in succession, stop, reverse, and set the ePlayer options.

### 2.4.3 Selecting Entries

Many operations in Maestro are performed on the entries selected in the Project Table. The Project Table functions much like any other table: select rows by clicking, shift-clicking, and control-clicking. However, because clicking in an editable cell of a selected row enters edit mode, you should click in the Row column to select entries. See Section 2.4.5 on page 18 for more information on mouse actions in the Project Table. There are shortcuts for selecting classes of entries on the Select menu.

In addition to selecting entries manually, you can select entries that meet a combination of conditions on their properties. Such combinations of conditions are called *filters*. Filters are Entry Selection Language (ESL) expressions and are evaluated at the time they are applied. For example, if you want to set up a Glide job that uses ligands with a low molecular weight (say, less than 300) and that has certain QikProp properties, you can set up a filter and use it to select entries for the job. If you save the filter, you can use it again on a different set of ligands that meet the same selection criteria.

**To create a filter:**

1. Do one of the following:
   - Choose Only, Add, or Deselect from the Select menu.
   - Click the Entry selection button on the toolbar.
2. In the Properties folder, select a property from the property list, then select a condition.
3. Combine this selection with the current filter by clicking Add, Subtract, or Intersect. These buttons perform the Boolean operations OR, AND NOT, and AND on the corresponding ESL expressions.
4. To save the filter for future use click Create Filter, enter a name, and click OK.
5. Click OK to apply the filter immediately.

### 2.4.4 Including Entries in the Workspace

In addition to selecting entries, you can also use the Project Table to control which entries are displayed in the Workspace. An entry that is displayed in the Workspace is *included* in the Workspace; likewise, an entry that is not displayed is *excluded*. Included entries are marked by an X in the diamond in the In column; excluded entries are marked by an empty diamond. Entry inclusion is completely independent of entry selection.

To include or exclude entries, click, shift-click, or control-click in the In column of the entries, or select entries and choose Include or Exclude from the Entry menu. Inclusion with the mouse works just like selection: when you include an entry by clicking, all other entries are excluded.

It is sometimes useful to keep one entry in the Workspace and include others one by one: for example, a receptor and a set of ligands. You can fix the receptor in the Workspace by selecting it in the Project Table and choosing Fix from the Entry menu or by pressing CTRL+F. A padlock icon replaces the diamond in the In column to denote a *fixed* entry. To remove a fixed entry from the Workspace, you must exclude it explicitly (CTRL+X). It is not affected by the inclusion or exclusion of other entries. Fixing an entry affects only its inclusion; you can still rotate, translate, or modify the structure.

### 2.4.5 Mouse Functions in the Project Table

The Project Table supports the standard use of shift-click and control-click to select objects. This behavior applies to the selection of entries and the inclusion of entries in the Workspace. You can also drag to resize rows and columns and to move rows.

You can drag a set of non-contiguous entries to reposition them in the Project Table. When you release the mouse button, the entries are placed after the first unselected entry that precedes the entry on which the cursor is resting. For example, if you select entries 2, 4, and 6, and release the mouse button on entry 3, these three entries are placed after entry 1, because entry 1 is the first unselected entry that precedes entry 3. To move entries to the top of the table, drag them above the top of the table; to move entries to the end of the table, drag them below the end of the table.

A summary of mouse functions in the Project Table is provided in Table 2.3.

*Table 2.3. Mouse operations in the Project Table.*

| Task | Mouse Operation |
|---|---|
| Change a Boolean property value | Click repeatedly in a cell to cycle through the possible values (On, Off, Clear) |
| Display the Entry menu for an entry | Right-click anywhere in the entry. If the entry is not selected, it becomes the selected entry. If the entry is selected, the action is applied to all selected entries. |
| Display a version of the Property menu for a property | Right-click in the column header |
| Edit the text or the value in a table cell | Click in the cell and edit the text or value |
| Include an entry in the Workspace, exclude all others | Click the In column of the entry |
| Move selected entries | Drag the entries |
| Paste text into a table cell | Middle-click |
| Resize rows or columns | Drag the boundary with the middle mouse button |
| Select an entry, deselect all others | For an unselected entry, click anywhere in the row except the In column; for a selected entry, click the row number. |
| Select or include multiple entries | Click the first entry then shift-click the last entry |
| Toggle the selection or inclusion state | Control-click the entry or the In column |

### 2.4.6 Project Table Shortcut Keys

Some frequently used project operations have been assigned shortcut key combinations. The shortcuts, their functions, and their menu equivalents are listed in Table 2.4.

*Table 2.4. Shortcut keys in the Project Table.*

| Keys | Action | Equivalent Menu Choices |
|---|---|---|
| CTRL+A | Select all entries | Select > All |
| CTRL+F | Fix entry in Workspace | Entry > Fix |
| CTRL+I | Open Import panel | Table > Import Structures |
| CTRL+N | Include only selected entries | Entry > Include Only |
| CTRL+U | Deselect all entries | Select > None |
| CTRL+X | Exclude selected entries | Entry > Exclude |
| CTRL+Z | Undo/Redo last command | Edit > Undo/Redo in main window |

## 2.5 Building a Structure

After you start Maestro, the first task is usually to create or import a structure. You can open existing Maestro projects or import structures from other sources to obtain a structure, or you can build your own. To open the Build panel, do one of the following:

- Click the Open/Close Build panel button in the toolbar:
- Choose Build from the Edit menu.
- Press CTRL+B.

The Build panel allows you to create structures by drawing or placing atoms or fragments in the Workspace and connecting them into a larger structure, to adjust atom positions and bond orders, and to change atom properties. This panel contains a toolbar and three folders.

### 2.5.1 Placing and Connecting Fragments

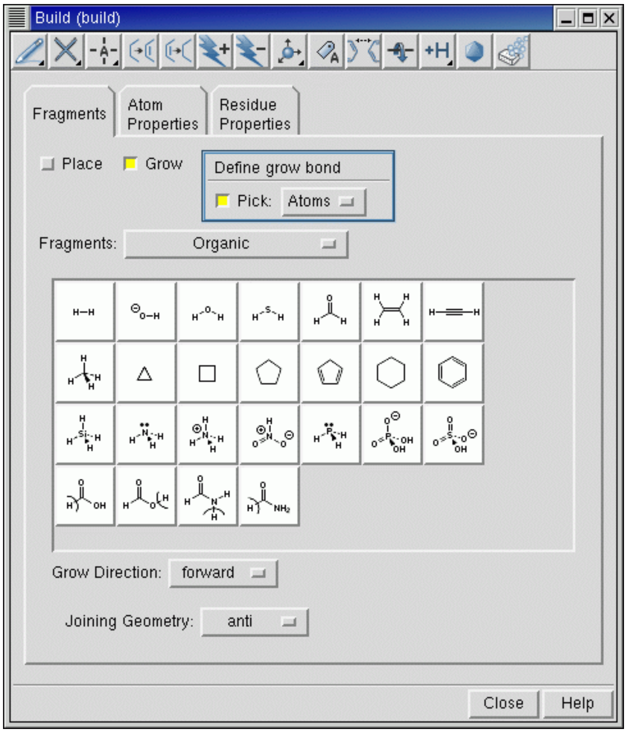
The Build panel provides several tools for creating structures in the Workspace. You can place and connect fragments, or you can draw a structure freehand.

**To place a fragment in the Workspace:**

1. Select Place.
2. Choose a fragment library from the Fragments menu.
3. Click a fragment.
4. Click in the Workspace where you want the fragment to be placed.

**To connect fragments in the Workspace, do one of the following:**

- Place another fragment and connect them using the Connect & Fuse panel, which you open from the Edit menu on the main menu bar or with the Display Connect & Fuse panel on the Build toolbar.
- Replace one or more atoms in the existing fragment with another fragment by selecting a fragment and clicking in the Workspace on the main atom to be replaced.
- Grow another fragment by selecting Grow in the Build panel and clicking the fragment you want to add in the Fragments folder.



Place [✓ Grow]

Define grow bond

[✓ Pick:] Atoms

Fragments: Organic

Grow Direction: forward

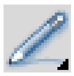
Joining Geometry: anti

***Figure 2.4. The* Build *panel.***

Grow mode uses predefined rules to connect a fragment to the *grow bond*. The grow bond is marked by a green arrow. The new fragment replaces the atom at the head of the arrow on the grow bond and all atoms attached to it. To change the grow bond, choose Bonds from the Pick option menu in the Build panel and click on the desired grow bond in the Workspace. The arrow points to the atom nearest to where you clicked.

**To draw a structure freehand:**

1. Choose an element from the Draw button menu on the Build panel toolbar:



2. Click in the Workspace to place an atom of that element.
3. Click again to place another atom and connect it to the previous atom.
4. Continue this process until you have drawn the structure.
5. Click the active atom again to finish drawing.

### 2.5.2 Adjusting Properties

In the Atom Properties folder, you can change the properties of the atoms in the Workspace. For each item on the Property option menu—Element, Atom Type (MacroModel), Partial Charge, PDB Atom Name, Grow Name, and Atom Name—there is a set of tools you can use to change the atom properties. For example, the Element tools consist of a periodic table from which you can choose an element and select an atom to change it to an atom of the selected element.

Similarly, the Residue Properties folder provides tools for changing the properties of residues: the Residue Number, the Residue Name, and the Chain Name.

To adjust bond lengths, bond angles, dihedral angles, and chiralities during or after building a structure, use the Adjust distances, angles or dihedrals button on the main toolbar:

You can also open the Adjust panel from this button menu, from the Display Adjust panel button on the Build panel toolbar (which has the same appearance as the above button) or from the Edit menu in the main window.

### 2.5.3 The Build Panel Toolbar

The toolbar of the Build panel provides quick access to tools for drawing and modifying structures and labeling atoms. See Section 2.3.2 on page 9 for a description of the types of toolbar buttons. The toolbar buttons and their use are described below.

Free-hand drawing
Choose an element for drawing structures freehand in the Workspace (default C). Each click in the Workspace places an atom and connects it to the previous atom.

Delete
Choose an object for deleting. Same as the Delete button on the main toolbar, see page 10.

Set element
Choose an element for changing atoms in the Workspace (default C). Click an atom to change it to the selected element.

Increment bond order
Select a bond to increase its bond order by one, to a maximum of 3.

Decrement bond order
Select a bond to decrease its bond order by one, to a minimum of 0.

Increment formal charge
Select an atom to increase its formal charge by one.

Decrement formal charge
Select an atom to decrease its formal charge by one.

Move
Choose a direction for moving atoms, then click the atom to be moved. Moves in the XY plane are made by clicking the new location. Moves in the Z direction are made in 0.5 Å increments.

Label
Apply heteroatom labels as you build a structure. The label consists of the element name and formal charge, and is applied to atoms other than C and H.

Display Connect & Fuse panel
Open the Connect & Fuse panel so you can connect structures (create bonds between structures) or fuse structures (replace atoms of one structure with those of another).

Display Adjust panel
Open the Adjust panel so you can change bond lengths, bond angles, dihedral angles, or atom chiralities.

Add hydrogens
Choose an atom type for applying the current hydrogen treatment. Same as the Add hydrogens button on the main toolbar, see page 10.

Geometry Symmetrizer
Open the Geometry Symmetrizer panel for symmetrizing the geometry of the structure in the Workspace.

Geometry Cleanup
Clean up the geometry of the structure in the Workspace.

## 2.6 Selecting Atoms

Maestro has a powerful set of tools for selecting atoms in a structure: toolbar buttons, picking tools in panels, and the Atom Selection dialog box. These tools allow you to select atoms in two ways:

- Select atoms first and apply an action to them
- Choose an action first and then select atoms for that action

### 2.6.1 Toolbar Buttons

The small triangle in the lower right corner of a toolbar button indicates that the button contains a menu. Many of these buttons allow you to choose an object type for selecting: choose Atoms, Bonds, Residues, Chains, Molecules, or Entries, then click on an atom in the Workspace to perform the action on all the atoms in that structural unit.

For example, to select atoms with the Workspace selection toolbar button:

1. Choose Residues from the Workspace selection button menu:

   The button changes to:

2. Click on an atom in a residue in the Workspace to select all the atoms in that residue.

## 2.6.2 Picking Tools

The picking tools are embedded in each panel in which you need to select atoms to apply an operation. The picking tools in a panel can include one or more of the following:

- Pick option menu—Allows you to choose an object type. Depending on the operation to be performed, you can choose Atoms, Bonds, Residues, Chains, Molecules, or Entries, then click on an atom in the Workspace to perform the action on all the atoms in that structural unit.

  The Pick option menu varies from panel to panel, because not all object types are appropriate for a given operation. For example, some panels have only Atoms and Bonds in the Pick option menu.
- All button—Performs the action on all atoms in the Workspace.
- Selection button—Performs the action on any atoms already selected in the Workspace.
- Previous button—Performs the action on the most recent atom selection defined in the Atom Selection dialog box.
- Select button—Opens the Atom Selection dialog box.
- ASL text box—Allows you to type in an ASL expression for selecting atoms.

  ASL stands for Atom Specification Language, and is described in detail in the *Maestro Command Reference Manual*.
- Clear button—Clears the current selection
- Show markers option—Marks the selected atoms in the Workspace.

For example, to label atoms with the Label Atoms panel:

1. Choose Atom Labels from the Display menu.
2. In the Composition folder, select Element and Atom Number.
3. In the picking tools section at the top of the panel, you could do one of the following:
    - Click Selection to apply labels to the atoms already selected in the Workspace (from the previous example).
    - Choose Residues from the Pick option menu and click on an atom in a different residue to label all the atoms in that residue.

### 2.6.3 The Atom Selection Dialog Box

If you wish to select atoms based on more complex criteria, you can use the Atom Selection dialog box. To open this dialog box, choose Select from a button menu or click the Select button in a panel. See Section 5.3 of the *Maestro User Manual* for detailed instructions on how to use the Atom Selection dialog box.

## 2.7 Scripting in Maestro

Although you can perform nearly all Maestro-supported operations through menus and panels, you can also perform operations using Maestro commands, or compilations of these commands, called *scripts*. Scripts can be used to automate lengthy procedures or repetitive tasks and can be created in several ways. These are summarized below.

### 2.7.1 Python Scripts

Python is a full-featured scripting language that has been embedded in Maestro to extend its scripting facilities. The Python capabilities within Maestro include access to Maestro functionality for dealing with chemical structures, projects, and Maestro files.

The two main Python commands used in Maestro are:

- `pythonrun`—executes a Python module. (You can also use the alias `pyrun`.) The syntax is:

    `pythonrun` *module.function*

- `pythonimport`—rereads a Python file so that the next time you use the `pythonrun` command, it uses the updated version of the module. (You can also use the alias `pyimp`.)

From the Maestro Scripts menu you can install, manage, and run Python scripts. For more information on the Scripts menu, see Section 13.1 of the *Maestro User Manual*.

For more information on using Python with Maestro, see *Maestro Scripting with Python*.

## 2.7.2 Command Scripts

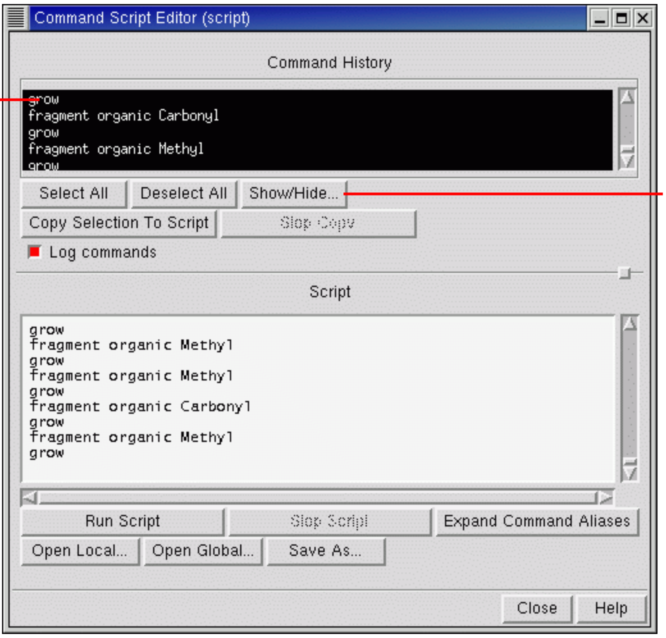
All Maestro commands are logged and displayed in the Command Script Editor panel. This means you can create a command script by performing the operations with the GUI controls, copying the logged commands from the Command History list into the Script text area of the panel, then saving the list of copied commands as a script.

**To run an existing command script:**

1. Open the Command Script Editor panel from the Edit menu in the main window.
2. Click Open Local and navigate to the directory containing the desired script.
3. Select a script in the Files list and click Open.

   The script is loaded into the Script window of the Command Script Editor panel.
4. Click Run Script.

Command scripts cannot be used for Prime operations.


The *Command History* window displays a log of all commands issued internally within Maestro when you interact with a panel, menu, or structure
Command Script Editor (script)
Command History
Select All
Deselect All
Show/Hide...
Copy Selection To Script
Stop Copy
Log commands
Script
Run Script
Stop Script
Expand Command Aliases
Open Local...
Open Global...
Save As...
Close
Help
Opens the *Show/ Hide Command* panel, used to determine which commands are logged in the *Command History* list

```
grow
fragment organic Carbonyl
grow
fragment organic Methyl
grow
```

```
grow
fragment organic Methyl
grow
fragment organic Methyl
grow
fragment organic Carbonyl
grow
fragment organic Methyl
grow
```

***Figure 2.5. The*** Command Script Editor ***panel.***

### 2.7.3 Macros

There are two kinds of macros you can create: named macros and macros assigned to function keys F1 through F12.

**To create and run a named macro:**

1. Open the Macros panel from the Edit menu in the main window.
2. Click New, enter a name for the macro, and click OK.
3. In the Definition text box, type the commands for the macro.
4. Click Update to update the macro definition.
5. To run the macro, enter the following in the command input area in the main window:

   `macrorun` *macro-name*

   If the command input area is not visible, choose Command Input Area from the Display menu.

**To create and run a function key macro:**

1. Open the Function Key Macros panel from the Edit menu in the main window.
2. From the Macro Key option, select a function key (F1 through F12) to which to assign the macro.
3. In the text box, type the commands for the macro.
4. Click Run to test the macro or click Save to save it.
5. To run the macro from the main window, press the assigned function key.

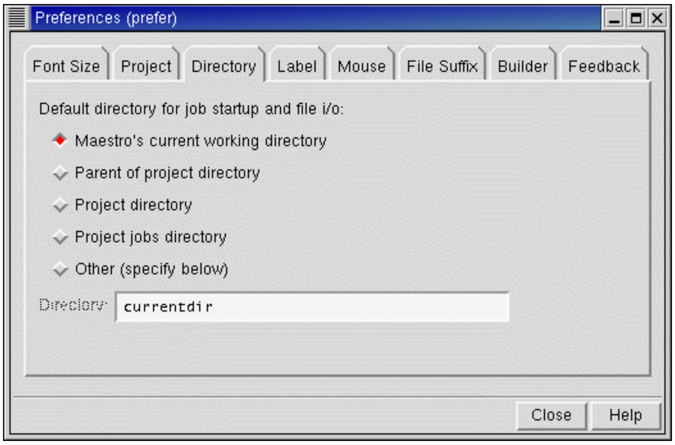
For more information on macros, see Section 13.5 of the *Maestro User Manual*.

## 2.8 Specifying a Maestro Working Directory

When you use Maestro to launch Prime jobs, Maestro writes job output to the directory specified in the Directory folder of the Preferences panel. By default, this directory (the file I/O directory) is the directory from which you started Maestro.

**To change the Maestro working directory:**

1. Open the Preferences panel from the Maestro menu.
2. Click the Directory tab.
3. Select the directory you want to use for reading and writing files.



Preferences (prefer)

Font Size | Project | Directory | Label | Mouse | File Suffix | Builder | Feedback

Default directory for job startup and file i/o:

[✓ Maestro's current working directory]

Parent of project directory

Project directory

Project jobs directory

Other (specify below)

Directory: currentdir

Close | Help

*Figure 2.6. T*

You can also set other preferences in the Preferences panel. See Section 12.2 of the *Maestro User Manual* for details.

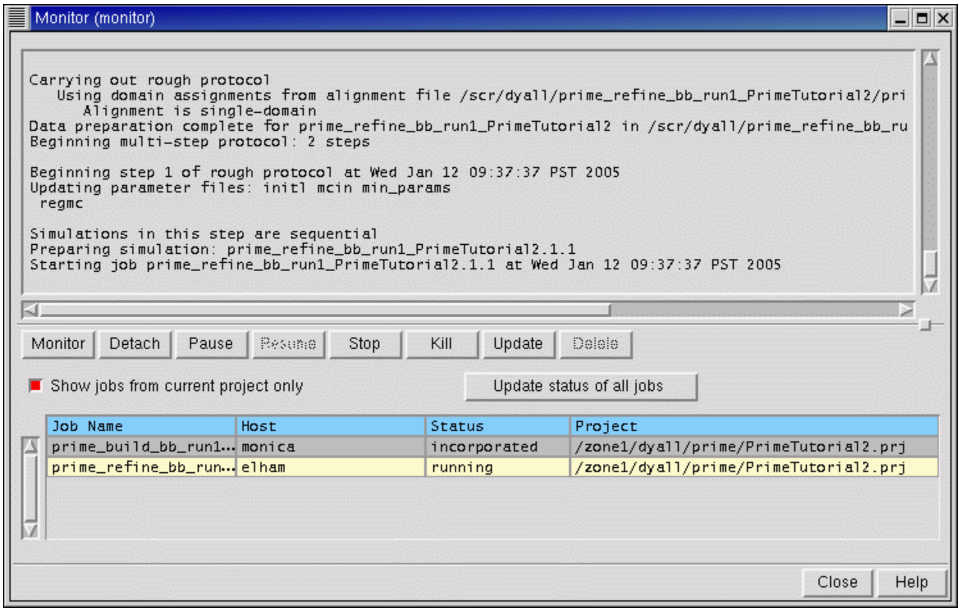
## 2.9 Undoing an Operation

To undo a single operation, click the Undo button in the toolbar, choose Undo from the Edit menu, or press CTRL+Z. The word Undo in the menu is followed by text that describes the operation to undo. Not all operations can be undone: for example, global rotations and translations are not undoable operations. For such operations you can use the Save view and Restore view buttons in the toolbar, which save and restore a molecular orientation.

## 2.10 Running and Monitoring Jobs

Maestro has panels for each product for preparing and submitting jobs. To use these panels, choose the appropriate product and task from the Applications menu and its submenus. Set the appropriate options in the panel, then click Start to open the Start dialog box and set options for running the job. For a complete description of the Start dialog box associated with your computational program, see your product's User Manual. When you have finished setting the options, click Start to launch the job and open the Monitor panel.

The Monitor panel is the control panel for monitoring the progress of jobs and for pausing, resuming, or killing jobs. All jobs that belong to your user ID can be displayed in the Monitor panel, whether or not they were started from Maestro. Subjobs are indented under their parent in the job list. The text pane shows various output information from the monitored job, such as the contents of the log file. The Monitor panel opens automatically when you start a job. If it is



```
Carrying out rough protocol
   Using domain assignments from alignment file /scr/dyall/prime_refine_bb_run1_PrimeTutorial2/pri
      Alignment is single-domain
Data preparation complete for prime_refine_bb_run1_PrimeTutorial2 in /scr/dyall/prime_refine_bb_ru
Beginning multi-step protocol: 2 steps

Beginning step 1 of rough protocol at Wed Jan 12 09:37:37 PST 2005
Updating parameter files: initl mcin min_params
 regmc

Simulations in this step are sequential
Preparing simulation: prime_refine_bb_run1_PrimeTutorial2.1.1
Starting job prime_refine_bb_run1_PrimeTutorial2.1.1 at Wed Jan 12 09:37:37 PST 2005
```

| Job Name | Host | Status | Project |
|---|---|---|---|
| prime_build_bb_run1… | monica | incorporated | /zone1/dyall/prime/PrimeTutorial2.prj |
| prime_refine_bb_run… | elham | running | /zone1/dyall/prime/PrimeTutorial2.prj |

***Figure 2.7. The*** Monitor ***panel.***

not open, you can open it by choosing Monitor from the Applications menu in the Maestro main window.

While jobs are running, the Detach, Pause, Resume, Stop, Kill, and Update buttons are active. When there are no jobs currently running, only the Monitor and Delete buttons are active. These buttons act on the selected job. By default, only jobs started from the current project are shown. To show other jobs, deselect Show jobs from current project only.

When a monitored job ends, the results are incorporated into the project according to the settings used to launch the job. If a job that is not currently being monitored ends, you can select it in the Monitor panel and click Monitor to incorporate the results. Monitored jobs are incorporated only if they are part of the current project. You can monitor jobs that are not part of the current project, but their results are not incorporated. To add their results to a project, you must open the project and import the results.

Further information on job control, including configuring your site, monitoring jobs, running jobs, and job incorporation, can be found in the *Job Control Guide* and the *Installation Guide*.

## 2.11 Getting Help

Maestro comes with automatic, context-sensitive help (Auto-Help), Balloon Help (tooltips), an online help facility, and a user manual. To get help, follow the steps below:

- Check the Auto-Help text box at the bottom of the main window. If help is available for the task you are performing, it is automatically displayed there. It describes what actions are needed to perform the task.
- If your question concerns a GUI element, such as a button or option, there may be Balloon Help for the item. Pause the cursor over the element. If the Balloon Help does not appear, check that Show Balloon Help is selected in the Help menu of the main window. If there is Balloon Help for the element, it appears within a few seconds.
- If you do not find the help you need using either of the steps above, click the Help button in the lower right corner of the appropriate panel. The Help panel is displayed with a relevant help topic.
- For help with a concept or action not associated with a panel, open the Help panel from the Help menu or press CTRL+H.

If you do not find the information you need in the Maestro help system, check the following sources:

- The *Maestro User Manual*
- The Frequently Asked Questions page, found at http://www.schrodinger.com/Support/faq.html

You can also contact Schrödinger by e-mail or phone for help:

- E-mail: help@schrodinger.com
- Phone: (503) 299-1150

## 2.12 Ending a Maestro Session

To end a Maestro session, choose Quit from the Maestro menu. To save a log file with a record of all operations performed in the current session, click Quit, save log file in the Quit panel. This information can be useful to Schrödinger support staff when responding to any problem you report.

# Using Prime–Structure Prediction

This chapter presents the essentials of using Prime–Structure Prediction (Prime–SP):

- Starting, naming, and saving Prime–SP runs in Maestro projects
- Navigating within and between runs
- Job control, job monitoring, and Prime–SP job options
- Using Prime–SP step panels, including their common layout and features
- Using the Prime menu bar and Prime toolbar

## 3.1 Prime Runs and Maestro Projects

A single execution of the Prime workflow using a particular set of choices of templates, paths, and settings is called a *run* .

Prime runs are stored within Maestro projects. Structures generated in completed Prime runs are added to the Maestro Project Table. If you are unfamiliar with Maestro projects and the Maestro Project Table, you may want to review Section 2.4 on page 13 and the Project Table and Project Facility online help topics. For more information, see the *Maestro User Manual*.

Maestro always has a project open. If you begin to work without selecting or naming a project, Maestro creates a *scratch* (unnamed) project.

Prime always has a run open, which belongs to the current project (whether it is a named project or a scratch project). If you begin the Prime workflow without specifying an existing run, the default name for the current run is `run1`. Multiple runs can be performed within a project, but you must save the project in order to save the data in the runs.

Each project can be displayed in the Project Table panel. In Prime, Project Table entries are usually finished model structures with their properties. Prime stores data in the project as each step in a run is performed, but most of this data is not displayed in the Project Table, which can remain empty until a run is completed. For this reason you should save scratch projects in which you have done work, even if the Project Table is empty.

Two Maestro projects can be merged into one. If you merge two Maestro projects, each of which contains one or more Prime runs, all the runs are included in the merged project. If two runs have the same name, Maestro automatically makes the names unique by appending `-mrg1` to one of them.

Maestro projects containing multiple runs are useful for comparing the results of workflows in which different selections and options were chosen: for example, a different template selected for building, or a job performed with a different set of options. Creating these multiple runs may involve navigating to an earlier step in your current run, making different selections, and then navigating forward again in a new run. For a description of the features used in navigation (the Next button, the Back button, and the Guide), see Section 3.2 on page 33.

If you return to an earlier step in your current run and make different choices as you proceed forward, you will be asked whether you want to overwrite your existing data. If you want to keep the data in your current run for comparison rather than overwriting it, you can start a new run. The commands for saving the old run, starting a new one, and switching between Prime runs are found under the Prime File menu.

The File menu commands for managing runs are:

New

Asks for a new run name (all run names must be unique), and starts a new run at the Input Sequence step.

Save As

Copies the current run under a new run name and switches to the new run. The original run is saved under the original name. If you open a new project and begin a workflow without explicitly giving the run a name, it will be given the default name of `run1`.

Rename

Asks for a new name for the current run. All run names must be unique.

Delete

Deletes the current run, on confirmation.

Open

Displays a list of options for switching to another run. The list includes the four most recently used runs. If there are more than four runs in the current project, click the More option to select a run by name.

**To backtrack and try a different set of choices without overwriting your current workflow:**

1. Choose Save As from the Prime File menu.

   This saves the current run under the old name (`run1` is the default for the first run in a new project) and creates a copy under the new name.

You are now in a new workflow identical to the first except for its name.

2. Navigate to the step where you want to make a different choice. If you want to go back more than one step, click the step name in the Guide diagram.

## 3.2 General Panel Layout

Each step in Prime–SP has its own panel. The panels have many elements in common and share a standard layout. This section discusses these common elements, their location, and their function. An example of a Prime–SP panel is shown in Figure 3.1.

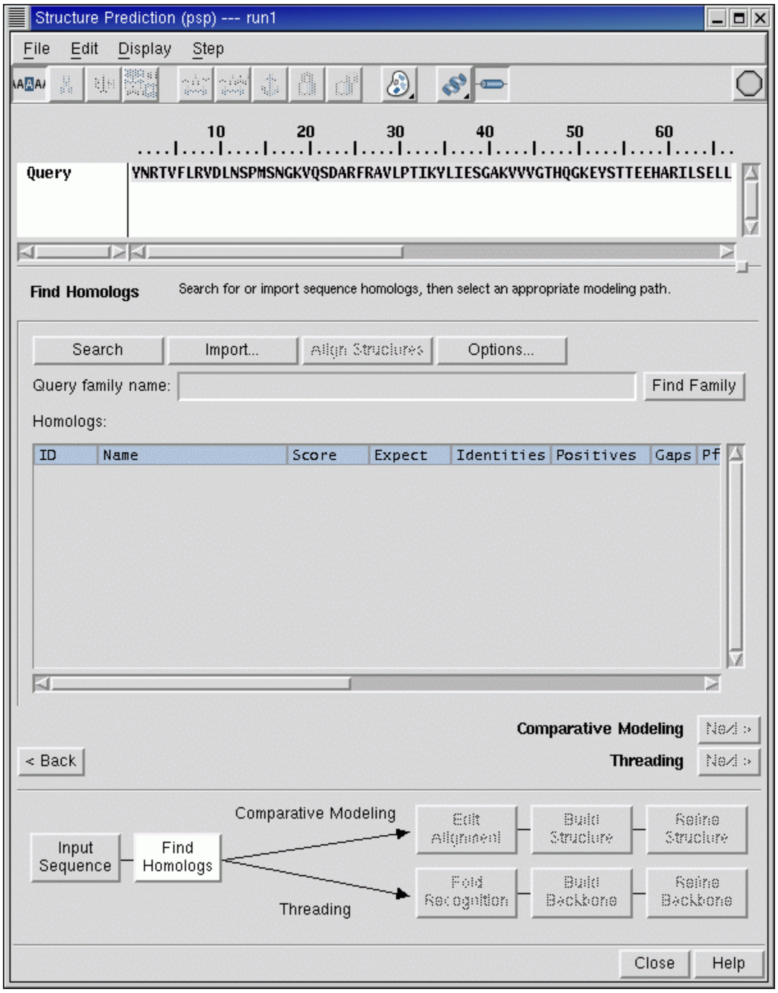


*Figure 3.1. The* Find Homologs *panel.*

In the upper part of the panel is the Prime menu bar, and below that is the Prime toolbar. Below the Prime toolbar is the Prime sequence viewer. You can resize the sequence viewer by dragging the small box in the right-hand corner of the horizontal line that separates the sequence viewer from the main panel.

In addition to displaying sequences for the query and for templates, in some steps the sequence viewer shows secondary structure predictions (SSPs). Right-click on an SSP for a menu of commands that includes Hide All, Delete, and Export. Right-clicking on a sequence also produces a shortcut menu of commands appropriate to the current step. You can export sequences using the Export Sequences command under the Prime File menu.

Between the sequence viewer and the toolbar is a “ruler”, which displays the residue position relative to the leftmost residue displayed in the sequence viewer. This residue position is displayed in parentheses in the tooltip for each residue. The positions are arbitrary, and can change when you display residues in the viewer. They are intended to help you keep track of the sequence as you scroll the sequence viewer.

Below the sequence viewer are the step name and a brief description of the step, followed by a row of buttons. The first button on the left runs the main action of the step, which may involve launching a job. See Section 3.5 on page 40 for more information on running Prime jobs. An Options button is usually located to the right of the action buttons. Most buttons, including those in the Prime toolbar, the Prime Guide, and the Maestro toolbar, have tooltips that you can view by pausing the cursor over the button.

The gray octagon in the upper right corner of the panel turns green and spins when you launch a job. When the octagon turns gray again, the job is finished. Click the octagon to open the Job Monitor panel. For more information on job control, see Section 3.5 on page 40.

The center of the panel contains the Step View, which shows data relevant to the step in table format. In the Build Structure step, the Step View contains output from the Build Structure job’s log file. You can view the complete text of a truncated table entry as a tooltip by pausing the cursor over the text. To resize a table column, drag a column border using the middle mouse button. You can sort the tables in the Step View by clicking the column heading, once for ascending order, twice for descending order.

Below the Step View are the Back and Next buttons, the primary way to navigate between steps. You can perform a simple workflow by clicking Next after completing each step. However, it is sometimes useful to go back to previous steps, make different choices, and proceed forward again using the Next button or the Guide, explained next.

The Guide, below the Back and Next buttons, is a diagram of the Prime workflow as a series of steps, each step represented by a button. The diagram branches to show the steps of both paths, Comparative Modeling and Threading.

To hide the Guide, deselect the Guide check box in the Step menu.

The Guide has two purposes: to show your current location in the Prime–SP process, and to let you navigate back and review or change the input at previous steps in the process. The current step is highlighted in the Guide diagram. All previously accessed steps, forward or backward, are available. You can go to any of these steps by clicking them in the Guide.

Clicking on an available step button in the Guide displays the panel for that step. If a step is not yet available, its button is dimmed. Available steps include those that have already passed through in the current run, the present step, and (once step-specific criteria have been met) the next step beyond the current location. When the next step is available, clicking on it behaves the same as clicking the Next button.

**Note:** You cannot launch searches and jobs by clicking the steps in the Guide. To launch actions pertaining to a step, click the appropriate buttons within the step panel.

The Close button appears below the Guide on the lower left of the panel. To the right is the Help button, which opens online help related to the current step.

## 3.3 The Prime Menu Bar

The Prime menu bar consists of four menus: File, Edit, Display, and Step. Some of the options in these menus are also available as toolbar buttons.

### 3.3.1 The File Menu

The commands on the File menu for working with Prime runs (New, Save As, Rename, Delete, and Open) are discussed in Section 3.1 on page 31. The menu also contains the commands Export Sequences and Job Options. See Section 3.5 on page 40 for more information on job options.

### 3.3.2 The Edit Menu

The Edit menu contains commands for editing sequences and secondary structure predictions. These editing tools are also available from the Prime toolbar, and are described with the toolbar in Section 3.4 on page 39.

### 3.3.3 The Display Menu

The Display menu includes the following commands for controlling the appearance of the sequence viewer and structures in the Maestro Workspace.

### Color Scheme

This command opens a menu of coloring options. In Prime–SP, the coloring schemes act as modes; that is, only one color scheme can be used at any given time. Each step in Prime has a default color scheme. You can use this menu to choose a different one. When amino acid sequences are added to the sequence viewer, they are colored according to the scheme that is currently selected.

When the structure associated with a sequence is displayed in the Workspace, it is generally colored using the same scheme as the sequence. However, protein structures in the Workspace are sometimes colored using other color schemes:

- When a PDB structure is imported, it is displayed in the Workspace using an error reporting color scheme in which complete recognized residues are gray, incomplete residues are red, unknown or nonstandard residues are orange, residues with multiple location indicators are green, and recognized residues with some unrecognized atoms are blue. For more information about conversion of PDB structures, see the *Maestro User Manual*.
- When the Build Structure step is complete, the color scheme applied to the structure in the Workspace is Atom PDB B Factor (Temperature Factor), in which blue atoms are those derived directly from the template, and red atoms have been predicted or modeled. To restore this color scheme, click the Maestro toolbar button Color all atoms by scheme

  and select Atom PDB B Factor (Temperature Factor) from the button menu.
- At the end of the Prime workflow, when an output structure is added to the Project Table, it is colored to represent the quality of the model. The final structural analysis provides information about potential problems: clashes, bad bond lengths, and bad bond angles. (Information from this analysis can also be viewed in the Protein Reporting panel.) Each atom in the structure is assigned a color, from blue (no problems) through green, yellow, and orange, to red (at least one severe problem.)

The Prime color schemes are:

- None: All sequences are colored standard gray.
- Color by Sequence: The query sequence is colored standard gray. Each of the other sequences uses a different color.
- Residue Type: Each amino acid sequence is colored according to the array of residue type colors.

- Residue Property: Each amino acid sequence is colored according to the array of residue property colors (fewer colors than residue type).
- Residue Matching: The query and templates are colored in a pairwise manner. Matching residues are colored according to their residue type, and nonmatching residues are colored standard gray. If more than one template is chosen, the query will have more colored residues because the effect is cumulative. If only the query sequence is shown, it is colored standard gray for all residues.
- Residue Homology: Aligned query and template residues that are identical (highly conserved) are colored red. Aligned query and template residues that are similar according to the BLOSUM62 scoring matrix (conserved) are colored yellow. Gray is used for residues with no homology.
- Secondary Structure: The query sequence is colored standard gray. All other AA sequences should have SSAs, which are colored according to the SSA colors.
- Template ID: Only available for the Comparative Modeling path. The query sequence is colored standard gray. The predicted structure is colored according to template ID for each region (using the cycle colors). Each template sequence is colored with its color, but only in the region that was used (the rest of the template is colored dark gray to indicate it is not used).
- Proximity: At any step in which the sequence viewer is displaying sequences that have associated structure (all steps except Input Sequence), color by proximity mode is available. When you select this color scheme, the current sequence colors do not change immediately. Instead, when the mouse is over a sequence that has structure, the cursor changes to indicate that proximity can be calculated. You can then select one or more contiguous residues in the sequence, and the system will color that sequence based on the proximity to the selected residues. No other sequences will have modified colors (even if they have structure too and are in proximity of the selected sequence).

  Proximity is calculated based on the shortest distance between any two atoms in two residues, not including the bonded C and N atoms in the residues next to the selected residues.

  To change the proximity cutoff, choose Preferences from the Display menu and change the number in the Proximity cutoff text box.

### Font Size

The font size of letters and numbers in the sequence viewer can be changed using this option. The font size options are:

- Small
- Medium
- Large
- Huge

View Structure

The options for showing structures in the Maestro Workspace include:

- All
- None
- Predicted Only

View SSA

This option controls whether secondary structure assignments are shown in the sequence viewer.

Legend

Choose this menu option to open the Legend panel, which includes the assignment of each color and symbol used in each color scheme and sequence representation. Click the Colors tab and select a Color scheme from the list, or click the Symbols tab and select a type of Sequence data, such as Pfam/HMMER or SSP. This option is also available when you right-click on a sequence in the sequence viewer.

Preferences

Choose this Display menu option to open the Proximity dialog box:

- Proximity cutoff: The value entered in this text box defines proximity for the Proximity color scheme. The default is 4.00 Å.

## 3.3.4 The Step Menu

The Step menu provides another way to navigate between steps in Prime. It includes the following options:

Guide

When this check box is selected, the Prime Guide is displayed (the default). Deselect the check box to hide the Guide.

Input Sequence

Click this option to go to the Input Sequence step.

Find Homologs

Click this option to go to the Find Homologs step.

There is an option to click to go to each step in Prime–SP. A red diamond is displayed next to the option to go to the current step. Options to go to steps that are not available appear dimmed.

## 3.4 The Prime Toolbar

The Prime toolbar is the row of buttons just under the Prime menu bar. The Prime toolbar provides convenient access to commonly used Prime tasks. Pause the mouse over a button to see its description. These task modes are also available as options in the Edit and Display menus. The toolbar buttons are shown below as they appear on the toolbar, with their names and functions.

Select (Edit menu): Select a region of the sequence to see the corresponding residues highlighted in the 3D Workspace.

Crop (Edit menu): The cropping tool, which can be used in the Build Structure step to select regions of the query sequence that are not to be used for building.

Edit SSP (Edit menu): Edit secondary structure predictions. Highlight a region of the SSP, and then type either `H` for $\alpha$-helix, `E` for $\beta$-strand, or `-` for loop.

Set Template Regions (Edit menu): If more than one template was selected in the Find Homologs step, use this button in the Build Structure step to proceed to the mode that allows selection of template regions for building.

Slide Freely (Edit menu): Use this button to edit the alignment between query and template. Selecting a residue and sliding to the right creates N-terminal gaps. Selecting a residue and sliding to the left creates C-terminal gaps.

Slide As Block (Edit menu): Use this button to edit the alignment between query and template. Selecting a residue and sliding to the right creates N-terminal gaps. Selecting a residue and sliding to the left slides all residues as a block to the left.

Add Anchors (Edit menu): Set alignment constraints by setting anchors at residue positions. To remove anchors, click on the same residue again.

Lock Gaps (Edit menu): To prevent gaps from being collapsed during manual editing, lock the gaps using this button. Locked gaps are indicated with a **-** symbol.

Unlock Gaps (Edit menu) To allow gaps to collapse during manual editing, unlock the gaps using this button. Unlocked gaps are indicated with a ~ symbol.

Color Scheme (Display menu): Select a color scheme for coloring sequences in the sequence viewer. If there is a structure in the 3D Workspace associated with the sequence, it will also be colored.

View Structure (Display menu): Use this button to toggle structures in the 3D Workspace on and off.

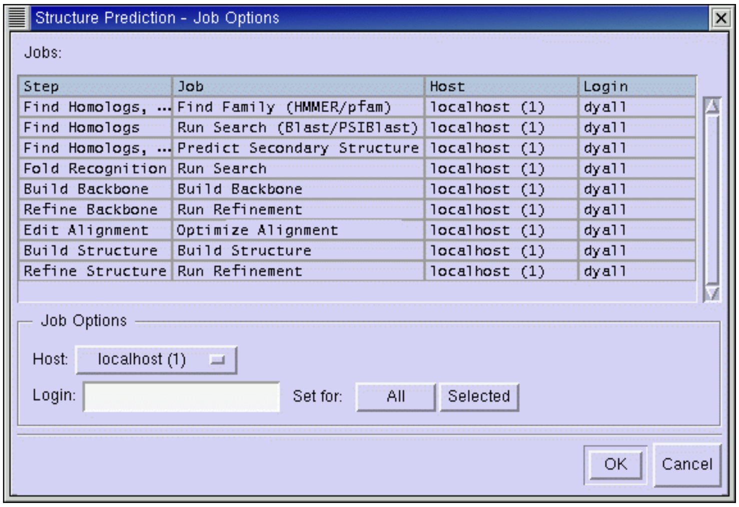
View SSA (Display menu): Use this button to toggle the secondary structure assignment (SSA) in the sequence viewer on and off.

## 3.5 Prime Job Options and Control

Before launching a Prime calculation from a step panel, you may want to choose which machine host the job will run on and check or modify job settings. When a run is complete, you can incorporate the results into the Maestro Project Table and save them as a project.

### 3.5.1 The Job Options Panel

The Structure Prediction–Job Options panel, shown in Figure 3.1, allows you to specify which machine each type of Prime job will run on, and under which login name. To open the panel, choose Job Options from the File menu on the Prime menu bar. The Job Options panel lists 10



| Step | Job | Host | Login |
|---|---|---|---|
| Find Homologs, … | Find Family (HMMER/pfam) | localhost (1) | dyall |
| Find Homologs | Run Search (Blast/PSIBlast) | localhost (1) | dyall |
| Find Homologs, … | Predict Secondary Structure | localhost (1) | dyall |
| Fold Recognition | Run Search | localhost (1) | dyall |
| Build Backbone | Build Backbone | localhost (1) | dyall |
| Refine Backbone | Run Refinement | localhost (1) | dyall |
| Edit Alignment | Optimize Alignment | localhost (1) | dyall |
| Build Structure | Build Structure | localhost (1) | dyall |
| Refine Structure | Run Refinement | localhost (1) | dyall |

***Figure 3.2. The*** Structure Prediction–Job Options ***panel.***

types of jobs, from Find Family (run from the Find Homologs step) to Run Refinement (run from the Refine Structure step). Specify the host machine and the login for each type of job. The default is `localhost`. Make sure your `schrodinger.hosts` file contains a list of available machines to run on. See the *Installation Guide* for information on the `schrodinger.hosts` file.

### 3.5.2 Setting Up and Launching Jobs

Though you specify the machines used to run jobs in the Job Options panel, you select Prime job settings and launch jobs from the Prime step panels. This can be as simple as clicking an action button, such as Search. In other steps, you select the type of job from a Task menu and then launch the job by clicking Run.

You can modify settings for step actions or tasks using the Options button. You select the residues or secondary structure features on which a refinement task will operate from lists, tables, and the Atom Selection dialog box in the Refine Structure step panel. Until you make a selection, the Run button remains unavailable.

For more information on setting up and launching jobs, see the discussions of specific steps in the chapters that follow.

### 3.5.3 Monitoring Jobs

After you launch a job, the gray octagon in the upper right corner of the step panel turns green and spins.

To monitor a job using the Maestro job Monitor panel, click the octagon. You can use this panel to monitor jobs from the current session or your previous sessions.

To kill a job, select the job name and then click Kill. If the job still appears to be running, you may have to go to the Project directory and remove the job file.

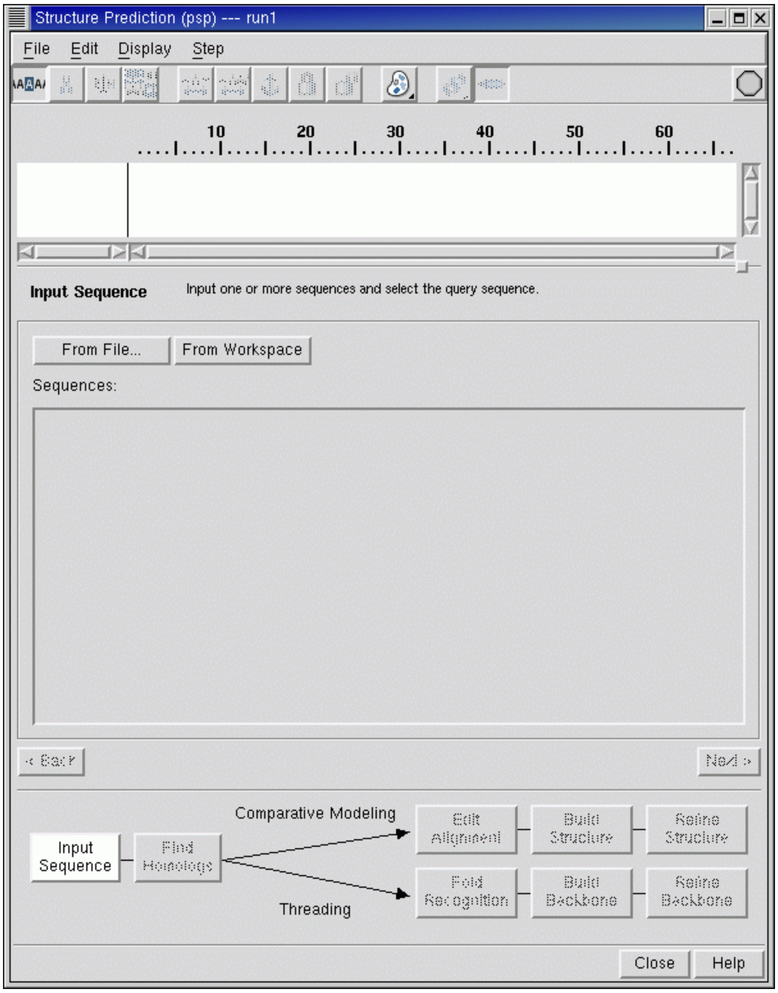
Various commands can be carried out from the Monitor panel (see Section 2.10 on page 28). For more information about the Monitor panel and monitoring jobs, see the *Maestro User Manual* and the online help.

### 3.5.4 Adding Structures to the Project Table

In the Refine Structure and Refine Backbone steps, predicted structures can be sent to the Maestro Project Table as new entries. This is useful for comparing predicted structures produced from different runs. Structures imported into Maestro can also be added as entries to the current Project Table.

# Prime–Structure Prediction: Initial Steps

This chapter outlines the initial steps in the Prime–SP workflow. In the Input Sequence step, a query sequence is imported. In the Find Homologs step, potential templates are identified, and a path is chosen: either Comparative Modeling or Threading. Later chapters describe the steps in each path in which a model is built and refined.



Input one or more sequences and select the query sequence.

*Figure 4.1.* ***The*** Input Sequence ***step, initial view.***

## 4.1 Input Sequence Step

This step is used to read in an amino acid sequence from a file or from a protein structure in the Workspace.

**To read a sequence from a file:**

- Click From File and specify a file containing the desired sequence.

  Several file formats are supported, including FASTA, EMBL, GENBANK, and PIR. For a complete list, see Appendix B. File formats are auto-detected by the program. Sequences must be in standard uppercase one-letter code.

**To read a sequence from the Workspace:**

- Click From Workspace to read sequences from proteins in the Workspace.

  If there are multiple chains with the same sequence, the sequence will be read in only once.

After input, the Sequences table in the Step View area shows all available unique sequences with their corresponding chain lengths in the order in which they are read. By default, the first sequence read in is displayed in the sequence viewer in the upper section of the panel. If multiple sequences have been read in, only one can be brought forward to the next step in a given Prime run.

**To select a query from the Sequences table:**

- Click on a sequence to select it for this run.

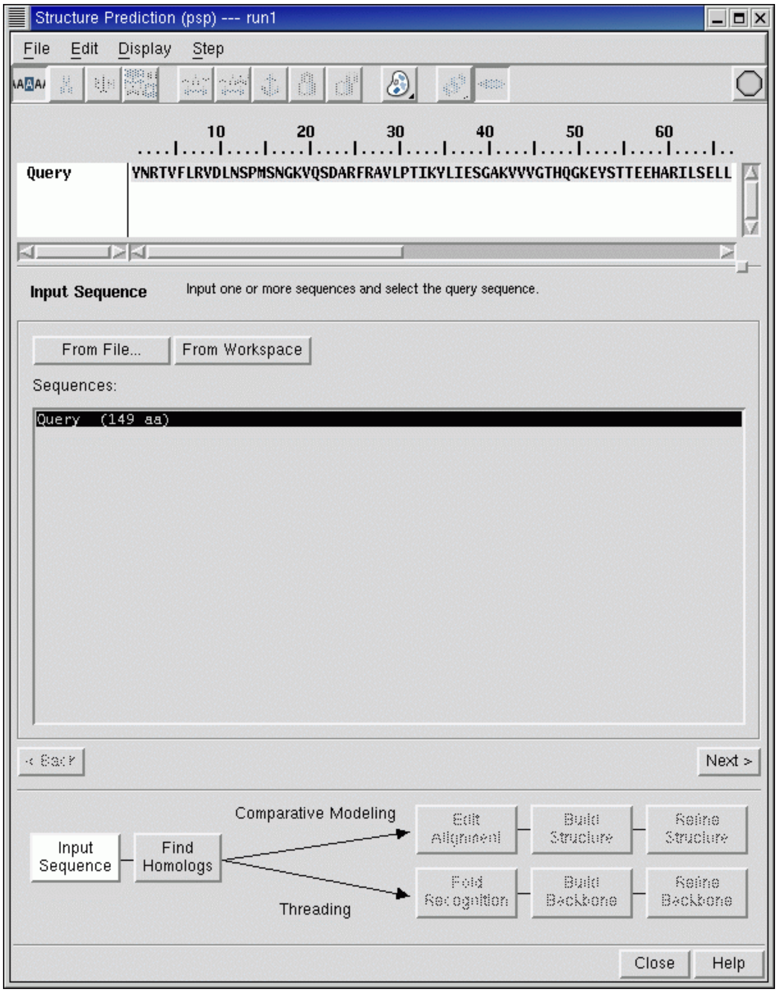
  This brings the selected sequence into the sequence viewer. The sequences are named by extracting the name from either the PDB header or the FASTA file header. Non-unique sequences are ignored. Non-unique names are modified to make them unique.

**Note:** Sequences longer than 1,000 residues generally include more than one domain. Divide these sequences into probable domains and treat each domain as an individual query.

**To continue:**

- When you are satisfied with the query sequence, click Next to go to the Find Homologs step.

  If an error message appears when you click Next, you cannot proceed until the problem is resolved. Error messages are listed in Appendix C. See Section C.1 on page 177 for help with Input Sequence error messages.



Query YNRTVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGKEYSTTEEHARILSELL

Input Sequence Input one or more sequences and select the query sequence.

Query (149 aa)

***Figure 4.2. The*** Input Sequence ***step, after input.***

## 4.2 Find Homologs Step

This step is used to search for homologs of the query sequence. These homologs can then be used as templates to build a model structure.

### 4.2.1 Importing Homologs

To import a homolog not in the NCBI sequence database, click Import. This button is also used to import homologs that have been rotated into a common reference frame when multiple templates have been selected.

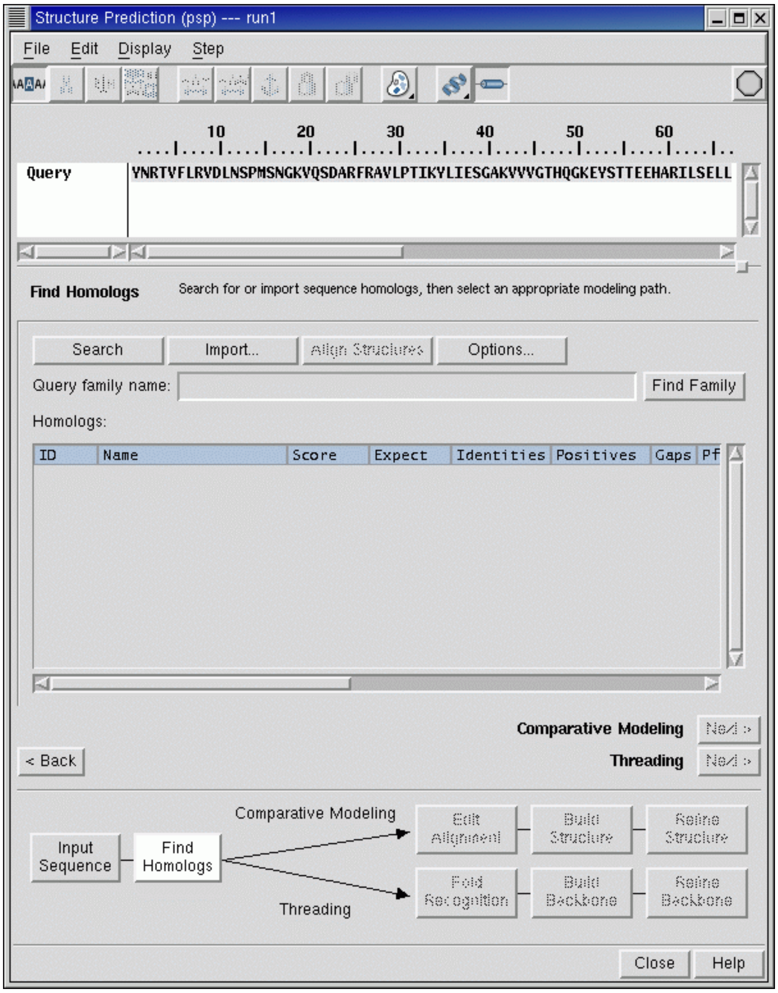


***Figure 4.3. The*** Find Homologs ***step, initial view.***

### 4.2.2 Searching for Homologs

**To launch a search for homologs:**

- Click Search (near the middle of the panel).

  Depending on what is selected in the Find Homologs–Options dialog box, BLAST or PSI-BLAST is used to search for templates. The default is to use BLAST to search the non-redundant PDB database that has been loaded with Prime. For search options, see Section 4.2.3.

**To identify a family for the query (optional):**

- Click Find Family to run HMMER/Pfam.

  This helps identify the family that best matches the query sequence. This job should take only a few minutes.

The query family name found appears in the Query family name text box with an E-value in parentheses. If the E-value of the family is too large, it is not meaningful to consider the family.

The sequence viewer shows the match between the query and the consensus sequence of the family. If this sequence, labeled *NAME*`_pfam`, is not visible, check for a plus sign to the left of the query name: +*NAME*. Click the plus sign to show any hidden sequences. The match sequence displays information about which residues are conserved in the multiple sequence alignment used to generate the Hidden Markov Model (HMM). Capital letters indicate highly conserved residues, lowercase letters indicate a match to the HMM, + means the match is conservative, and a blank indicates that the residue does not match the HMM.

### 4.2.3 Find Homologs Search Options

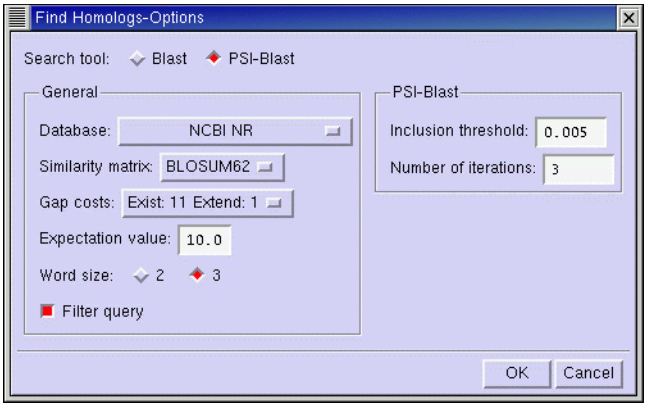
**To change search options:**

- Click Options to open the Find Homologs–Options dialog box.

  Use this dialog box to specify a PSI-BLAST or BLAST search, or change the settings for either type of search.

General options include:

- Database
- Similarity Matrix
- Gap Costs
- Expectation value
- Word size
- Filter query


Find Homologs-Options

Search tool: Blast [✓ PSI-Blast]

General

Database: NCBI NR

Similarity matrix: BLOSUM62

Gap costs: Exist: 11 Extend: 1

Expectation value: 10.0

Word size: 2 [✓ 3]

[✓ Filter query]

PSI-Blast

Inclusion threshold: 0.005

Number of iterations: 3

***Figure 4.4. The*** Find Homologs–Options ***dialog box.***

PSI-Blast options include:

- Inclusion threshold
- Number of iterations

The default database for searching is NCBI PDB (all). Other selections for the Database option include:

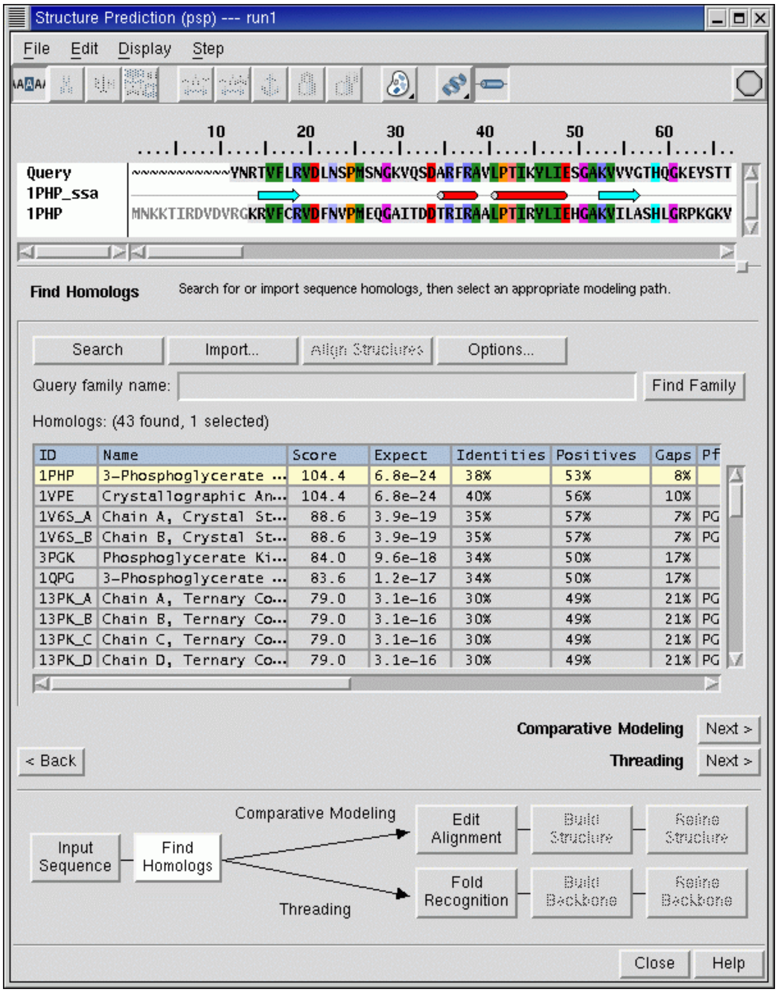
- NCBI PDB (non-redundant)—Does not include PDB files with identical sequences.
- NCBI NR—Recommended for PSI-BLAST runs. With this selection, only the sequences with structures available in the PDB will be displayed in the Homologs table.

## 4.2.4 Search Results

The output from the homolog search appears in the table labeled Homologs: (*N* found, *n* selected). For each potential template, the row contains the ID, the name, score, expectation value, % identities, % positives, % gaps, and header information (if taken from a PDB file). All percentages are rounded to the nearest whole number, which may be zero. A description of the table columns is given in Table 4.1.

You can view the complete text of each table cell as a tooltip by pausing the mouse over the cell. Columns can be sorted by clicking on the column heading. Click once to sort in descending order, and click twice to sort in ascending order. The column widths can be adjusted using the middle mouse button to click and drag the column border.

If no structural coordinates are available for a template, that row is dimmed and cannot be selected.



```
             10        20        30        40        50        60
Query    ~~~~~~~~~~~~YNRTVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGKEYSTT
1PHP_ssa
1PHP     MNKKTIRDVDVRGKRVFCRVDFNVPMEQGAITDDTRIRAALPTIRYLIEHGAKVILASHLGRPKGKV
```

Search for or import sequence homologs, then select an appropriate modeling path.

Query family name:

Homologs: (43 found, 1 selected)

| ID | Name | Score | Expect | Identities | Positives | Gaps | Pf |
|---|---|---|---|---|---|---|---|
| 1PHP | 3-Phosphoglycerate ... | 104.4 | 6.8e-24 | 38% | 53% | 8% | |
| 1VPE | Crystallographic An... | 104.4 | 6.8e-24 | 40% | 56% | 10% | |
| 1V6S_A | Chain A, Crystal St... | 88.6 | 3.9e-19 | 35% | 57% | 7% | PG |
| 1V6S_B | Chain B, Crystal St... | 88.6 | 3.9e-19 | 35% | 57% | 7% | PG |
| 3PGK | Phosphoglycerate Ki... | 84.0 | 9.6e-18 | 34% | 50% | 17% | |
| 1QPG | 3-Phosphoglycerate ... | 83.6 | 1.2e-17 | 34% | 50% | 17% | |
| 13PK_A | Chain A, Ternary Co... | 79.0 | 3.1e-16 | 30% | 49% | 21% | PG |
| 13PK_B | Chain B, Ternary Co... | 79.0 | 3.1e-16 | 30% | 49% | 21% | PG |
| 13PK_C | Chain C, Ternary Co... | 79.0 | 3.1e-16 | 30% | 49% | 21% | PG |
| 13PK_D | Chain D, Ternary Co... | 79.0 | 3.1e-16 | 30% | 49% | 21% | PG |

***Figure 4.5. The*** Find Homologs ***step with search results.***

If a chain ID is given for a homolog (e.g., Chain A in `1EMS_A`), then that chain, not the entire protein, is the potential template. When aligning multiple templates with multiple chains, you may need to extract single chains using the `getpdb` utility, as described in Section 14.1.

Once homologs are found, click on a row to view the structure of that template in the Maestro workspace. If no structural coordinates are available for selection, the row is dimmed. Selecting a template will also bring the sequence alignment of the template and the query sequence into the sequence viewer. Click on another template to deselect the first one and display the selection in the Workspace window.

*Table 4.1. Homologs table data*

| Column Header | Description |
|---|---|
| ID | The PDB ID code, including chain name. If a chain name is specified, then the chain, not the entire protein, is the potential template. |
| Name | Sequence name provided by BLAST. |
| Score | BLAST bit score. |
| Expect | BLAST expectation value. |
| Identities (%) | Percentage of residues that are identical between the sequences. |
| Positives (%) | Percentage of residues that are positive matches according to the similarity matrix selected in the Options dialog (default matrix is BLOSUM62). |
| Gaps (%) | Percentage of gaps in both query and homolog as returned by BLAST. |
| Title | From the PDB TITLE field. |
| Compound | From the PDB COMPND field. |
| Source | From the PDB SOURCE field. |
| Experiment | From the PDB EXPDTA field. This gives the experimental technique used to obtain the structure, e.g., X-RAY DIFFRACTION or NMR. |
| Resolution | From the PDB REMARK2 record, where applicable. |
| Ligands/Cofactors | From the PDB HET records: hetID (three alphanumeric characters) and name for each HET group. |
| Warning | A warning appears in this column for sequences that may be unusable for model building. See Section C.2.2, for a list of warnings. |

After you review the alignments, select the template that you want to build on by clicking on its row in the table. To select multiple templates, see Section 4.2.5.

## 4.2.5 Multiple Templates and Structure Alignment

To select more than one template for building, hold down the CTRL key and select another homolog in the table, or hold down the SHIFT key and select a set of homologs. If more than one template is selected, they will need to be *structurally aligned*, that is, rotated to bring them into a common reference frame, before they can be used in the next step.

- If you have already created a file with structurally aligned multiple templates, import it using the Import button, as described in Section 4.2.1.

- If you have not yet performed structural alignment on the selected templates, click the Align Structures button.

If structure alignment fails or does not give the intended result, the problem may be one of the following:

- The selected templates are structurally too dissimilar for a meaningful alignment.
- One or more of the template structures includes multiple chains, and either
    - Two or more templates are chains from one `.pdb` file (the structure alignment facility will not align chains from the same file), or
    - The chains that were aligned were not the chains you intended to align.

If the problem involves a template with multiple chains, you can use the `getpdb` utility (see Section 14.1) to extract each chain into a separate `.pdb` file, then run structure alignment again.

For information on running structure alignment from the command line, see Section 14.2.

## 4.2.6 Continuing to the Next Step

When you are satisfied with the template (or structurally aligned multiple templates) you have selected, follow the Comparative Modeling Path by proceeding to the Edit Alignment step.

If no satisfactory templates can be identified by BLAST/PSI-BLAST and you have not imported a template, you can follow the Threading Path by proceeding to the Fold Recognition step.

**Note:** The maximum length of a query in the Threading Path is 1,000 residues. It is recommended that such long sequences be divided into queries by probable domains, if this has not been done already, and the homolog search be repeated for these shorter queries before entering the Threading Path.

## 4.2.7 Saving Runs With Different Templates

If you decide later to try a different template, you can avoid overwriting your existing workflow by starting a new one before you navigate back to Find Homologs to choose a different homolog. Choose Save As from the File menu. This creates a copy of the current run, allowing you to select a new template and build on it. You can then compare results from each run.

### 4.2.8 Error Messages

If there is an error, a message appears when you click Next to go to the next step. See Section C.2 on page 177 for a list of error messages for Find Homologs.

# Comparative Modeling: Edit Alignment

## 5.1 The Comparative Modeling Path

In the initial steps of Prime–SP, you imported a query sequence (Input Sequence) and identified possible templates (Find Homologs). If a query-template match with a substantial percentage of identical residues has been selected, it is appropriate to continue the structure prediction process by following the Comparative Modeling path, a series of three Prime–SP steps used to build a model structure of the query sequence based on homology to a template or templates.

The structure prediction process in the Comparative Modeling path includes the following steps:

1. A satisfactory query-templates sequence alignment is produced in Edit Alignment.
2. The selected query-template alignment is used to build a model structure of the query in Build Structure.
3. The model structure undergoes selected energy-based optimization procedures in Refine Structure.

## 5.2 Overview of the Edit Alignment Step

In order to build a model structure of the query, the Build Structure step requires a satisfactory alignment between templates and query. The alignment of templates to query generated in the Find Homologs step is based only on sequence. Therefore, there is room for improvement. The Edit Alignment step allows you to improve alignments using various tools before building a model in the Build Structure step.

There are several ways to produce an alignment for use in Build Structure:

- Accept the BLAST/PSI-BLAST alignment generated in the Find Homologs step and currently displayed in the sequence viewer.
- Import an alignment.
- Generate or import secondary structure predictions, then use the Align program to create a new alignment.
- Edit any of these alignments manually.

If you want to search for a family that best matches the query sequence, but did not do so in the Find Homologs step, click the Find Family button to run HMMER/Pfam. For more information on this option, see Section 4.2.2 on page 47.

## 5.3 Initial and Imported Alignments

### 5.3.1 Accepting the BLAST/PSI-BLAST Alignment

When you start the Edit Alignment step, the BLAST/PSI-BLAST alignment for the template you selected in Find Homologs is displayed in the sequence viewer. See Figure 5.1.

If this is the template and the alignment you want to use, click Next and go to Build Structure. To accept a different template, using its existing alignment, click on the desired template and then click Next.

### 5.3.2 Exporting an Alignment

You may want to save the existing alignment for later use before making changes. Click the Export button to the right of the Alignments table to open the Export Alignments To File panel. To import the alignment again, see Section 5.3.3.

### 5.3.3 Importing an Alignment

Prime can also use an alignment generated in another program (or exported previously from Prime). The sequences in the alignment file must be exactly the same as those used by Prime, and so must the sequence and structure (PDB) names.

**Note:** The template PDB ID must be uppercase in the alignment file.

To import an alignment, click the Import button to the right of the Alignments table. This opens the file selection window (Import Alignments From File). For details of the allowed formats, see Appendix B.

## 5.4 Secondary Structure Predictions (SSPs)

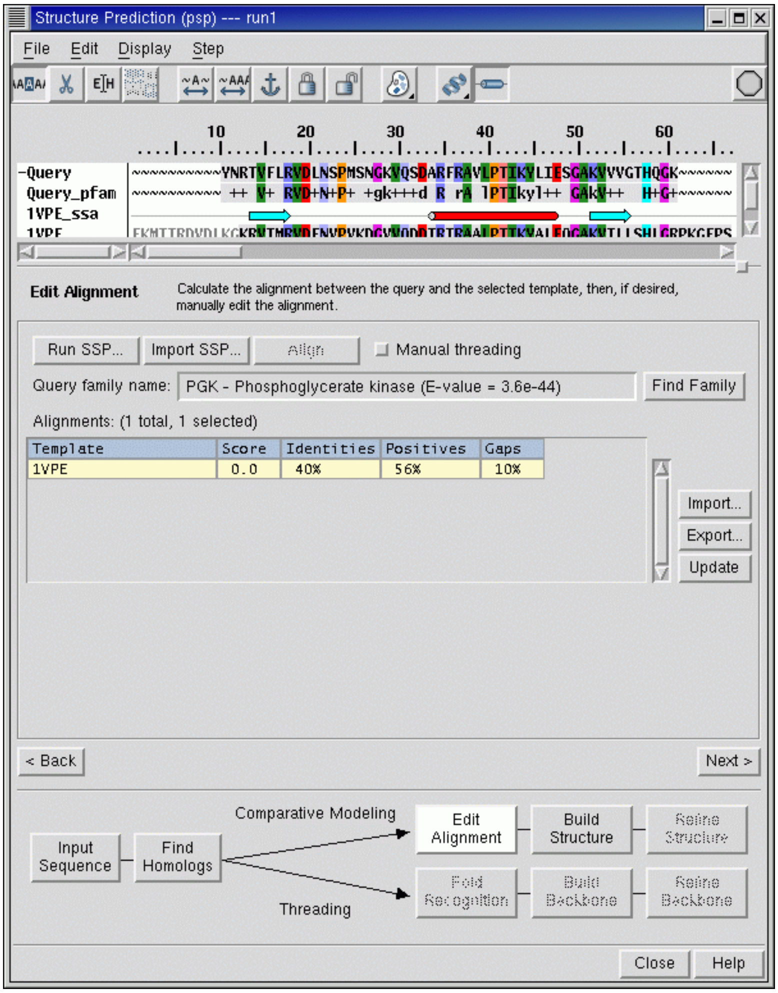
### 5.4.1 Generating SSPs

A secondary structure prediction for the query is required to run the Align program. To run all available SSP programs, click Run SSP. One secondary structure prediction program (SSpro) is bundled with Prime. However, two optional third-party programs are highly recommended for optimal results. See Chapter 15 for more information on these programs.

When the SSP run is finished, the secondary structure predictions are displayed in the sequence viewer below the query sequence. If there is a problem running the structure prediction programs, an error message is displayed.

## 5.4.2 Exporting an SSP

You may want to save an SSP for later use. Right-click on an SSP for a menu of options, including deleting the SSP, hiding all SSPs, or opening the Export SSP panel. By default, the SSP is saved in `m2io` (native Schrödinger) format. If there is a problem exporting a file, an error message is displayed.



```
         10        20        30        40        50        60
-Query       ~~~~~~~~~~YNRTVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGK~~~~~~
 Query_pfam  ~~~~~~~~~~ ++ V+ RVD+N+P+ +gk+++d R rA lPTIkyl++ GAkV++  H+G+~~~~~~
 1VPE_ssa
 1VPE        FKMTTRDVDLKGKRVTMRVDFNVPVKDGVVQDDTRIRAALPTIKYALEQGAKVILLSHLGRPKGEPS
```

Calculate the alignment between the query and the selected template, then, if desired, manually edit the alignment.

Query family name: PGK - Phosphoglycerate kinase (E-value = 3.6e-44)

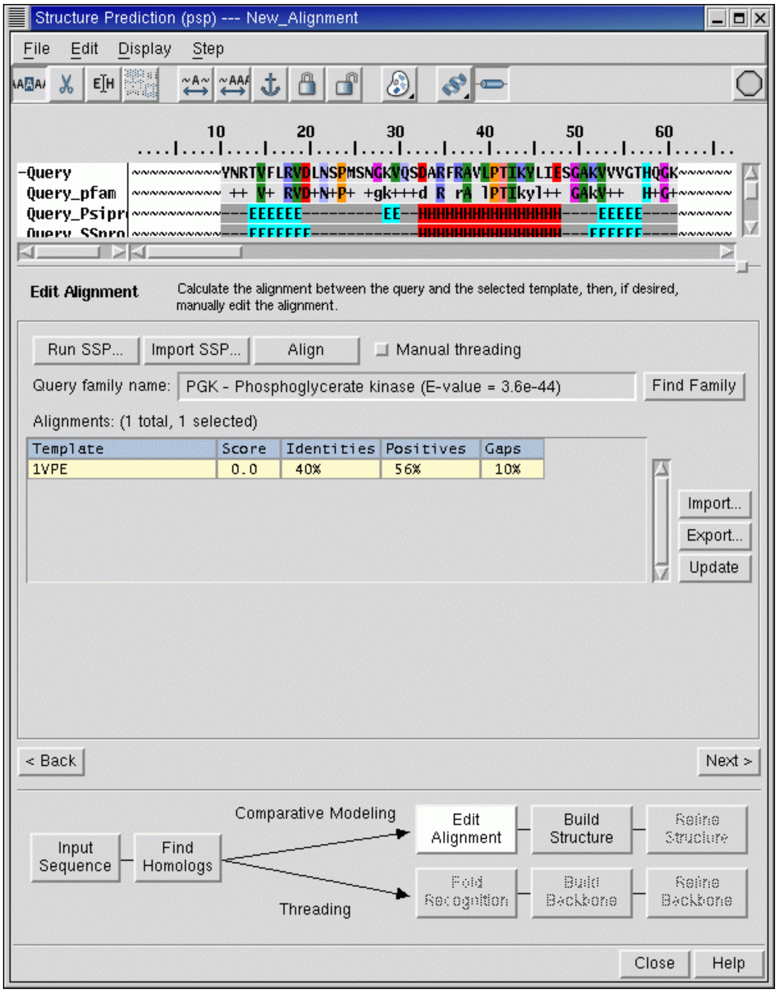
Alignments: (1 total, 1 selected)

| Template | Score | Identities | Positives | Gaps |
|---|---|---|---|---|
| 1VPE | 0.0 | 40% | 56% | 10% |

***Figure 5.1. The*** Edit Alignment ***panel, initial view.***

### 5.4.3 Importing an SSP

To import a secondary structure prediction for the query, click Import SSP and select the SSP file, which may be in FASTA or Maestro format. If necessary, use the `seqconvert` utility to change the file format. (See Section 14.4 for more information on `seqconvert`.) If the file contains bad data or the sequence doesn't match the query sequence, an error message is displayed.



```
-Query        ~~~~~~~~~~~YNRTVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGK~~~~~~
 Query_pfam   ~~~~~~~~~~~ ++ V+ RVD+N+P+ +gk+++d R rA lPTIkyl++ GAkV++   H+G+~~~~~~
 Query_Psipr  ~~~~~~~~~~~---EEEEEE---------EE--HHHHHHHHHHHHHHHH----EEEEE----~~~~~~
 Query_SSpro  ~~~~~~~~~~~---EEEEEEE-----------HHHHHHHHHHHHHHHH---EEEEEE----~~~~~~
```

**Edit Alignment** Calculate the alignment between the query and the selected template, then, if desired, manually edit the alignment.

Run SSP... | Import SSP... | Align | Manual threading

Query family name: PGK - Phosphoglycerate kinase (E-value = 3.6e-44) | Find Family

Alignments: (1 total, 1 selected)

| Template | Score | Identities | Positives | Gaps |
|---|---|---|---|---|
| 1VPE | 0.0 | 40% | 56% | 10% |

Import... | Export... | Update

< Back | Next >

*Figure 5.2. The* Edit Alignment *panel, after* Run SSP*.*

### 5.4.4 Deleting or Editing an SSP

If you believe an SSP is incorrect, you may delete it entirely or edit the incorrect portions. To delete an SSP, right-click on the SSP and choose Delete from the shortcut menu.

**To edit an SSP:**

1. Click the Edit SSP toolbar button:

2. Highlight the part of the SSP you want to change.
3. Type the character `e`, `h`, or `-`.

   E represents strand, H represents helix, and – represents loop.

The highlighted region of the prediction is changed to your choice.

### 5.4.5 Resetting SSPs

To retrieve an unmodified SSP after editing, click Run SSP. This immediately resets the SSP, but does not rerun the prediction programs.

## 5.5 Generated and Modified Alignments

### 5.5.1 Running the Align Program

When you are satisfied with the secondary structure predictions for the query, click Align to start the alignment job.

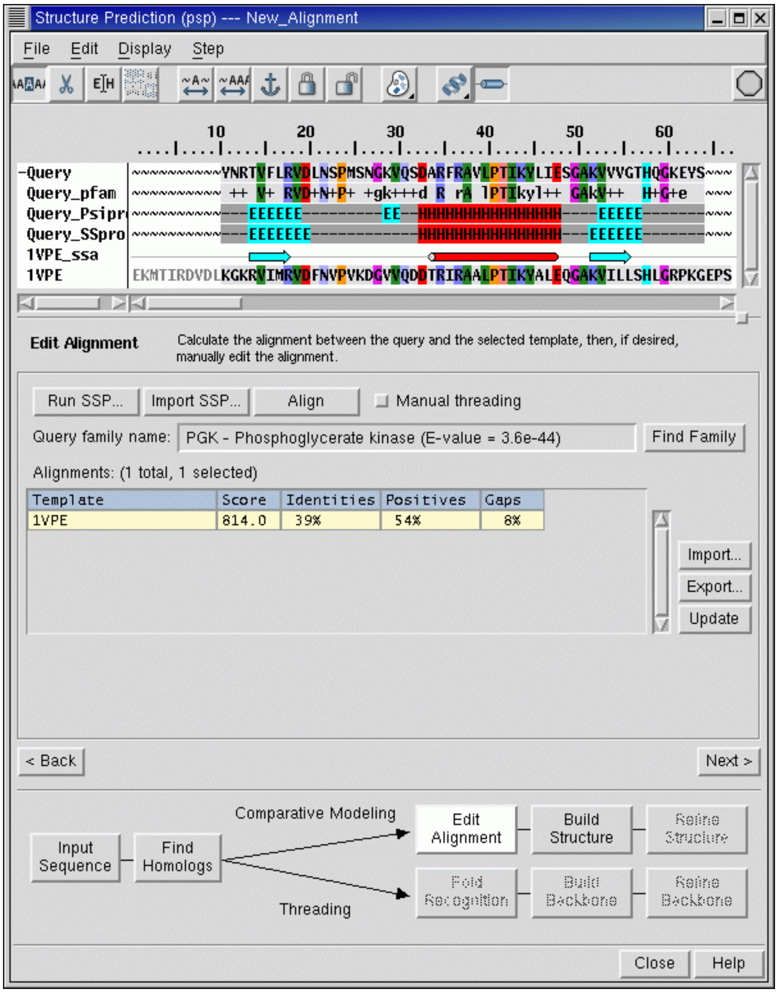
To take advantage of expert knowledge, one or more pairs of query/template residues that you know should be aligned can be constrained to remain so, by using alignment constraints.

**To constrain a pair of residues to remain aligned:**

1. Click the Add Anchors toolbar button.

2. Select one of the residues in the pair that you want to remain aligned during the Align calculation.

   An anchor symbol is displayed in the ruler at the corresponding residue position.



```
                 10        20        30        40        50        60
             ....|....|....|....|....|....|....|....|....|....|....|....|..
-Query        ~~~~~~~~~~YNRTVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGKEYS~~~
 Query_pfam   ~~~~~~~~~~ ++ V+ RVD+N+P+ +gk+++d R rA lPTIkyl++ GAkV++  H+G+e  ~~~
 Query_Psipr  ~~~~~~~~~~---EEEEEE--------EE--HHHHHHHHHHHHHHHH----EEEEE-------~~~
 Query_SSpro  ~~~~~~~~~~---EEEEEEE-----------HHHHHHHHHHHHHHHH---EEEEEE-------~~~
 1VPE_ssa
 1VPE         EKMTIRDVDLKGKRVIMRVDFNVPVKDGVVQDDTRIRAALPTIKYALEQGAKVILLSHLGRPKGEPS
```

**Edit Alignment** Calculate the alignment between the query and the selected template, then, if desired, manually edit the alignment.

Query family name: PGK - Phosphoglycerate kinase (E-value = 3.6e-44)

Alignments: (1 total, 1 selected)

| Template | Score | Identities | Positives | Gaps |
|---|---|---|---|---|
| 1VPE | 814.0 | 39% | 54% | 8% |

***Figure 5.3. The*** Edit Alignment ***panel after alignment.***

When Align completes the generation of the new alignment, you may accept it and continue to the next step or edit it manually as described below.

## 5.5.2 Align Program Technical Details

The goal of the program launched from the Edit Alignment step, Single Template Alignment, is to generate an accurate alignment between proteins with medium to high sequence identity (20-90%). Features of the program include:

- A position-specific substitutional matrix (PSSM) for the query sequence, derived from

PSI-BLAST, is used to match the template sequence.

- In order to minimize the inaccuracy in a single secondary structure prediction, a novel and robust algorithm has been designed to derive a composite secondary structure for the query sequence from multiple predictions. The composite SSP is aligned to the SSA of the template.

## 5.5.3 Manually Editing the Alignment

You can edit any alignment, however obtained, using combinations of these Prime toolbar buttons: Slide Freely, Slide as Block, Add(/Remove) Anchors, Lock Gaps, and Unlock Gaps.

Slide Freely (Edit menu): Use this button to edit the alignment between query and template. Selecting a residue and sliding to the right creates N-terminal gaps. Selecting a residue and sliding to the left creates C-terminal gaps.

Slide As Block (Edit menu): Use this button to edit the alignment between query and template. Selecting a residue and sliding to the right creates N-terminal gaps. Selecting a residue and sliding to the left slides all residues as a block to the left.

Add Anchors (Edit menu): Set alignment constraints by setting anchors at residue positions. To remove anchors, click on the same residue again.

Lock Gaps (Edit menu): To prevent gaps from being collapsed during manual editing, lock the gaps using this button. Locked gaps are indicated with a **-** symbol.

Unlock Gaps (Edit menu): To allow gaps to collapse during manual editing, unlock the gaps using this button. Unlocked gaps are indicated with a **~** symbol.

To assist in this task, you can view a preliminary model of the query in the Workspace. Select the Manual Threading toggle and click the Update button. The Workspace shows the template structure colored by query residue property. Each template residue is colored according to the property of the corresponding query residue in the alignment, revealing the location of charged, polar, and hydrophobic query residues. Use this option to make sure that insertions/gaps are not in the middle of regions of secondary structure such as alpha helices or beta sheets. It can also be used to check that hydrophobic residues point into hydrophobic regions and polar residues point towards solvent. As you make changes to the alignment, click Update again to see how each change has altered the mapping of the query's residue properties onto the template structure.

If there are obvious errors in the alignment (such as gaps in regions of secondary structure), a warning will appear in the Build Structure dialog box. Manual editing can be used to adjust the alignment before attempting to build the structure again.

### 5.5.4 Updating the Step View Table

After the Align job is complete, the Step View table is updated with the new alignment score, the percent identity, the percent similarity, and the percent gaps (all rounded to the nearest integer). This information can be updated after any change, including manual editing of the alignment, by clicking the Update button. The changed alignment is also reflected in the Maestro Workspace display of the structure.

## 5.6 Continuing to the Next Step

When you are satisfied with the alignment, click the Next button to proceed to the Build Structure step.

## 5.7 Error Messages

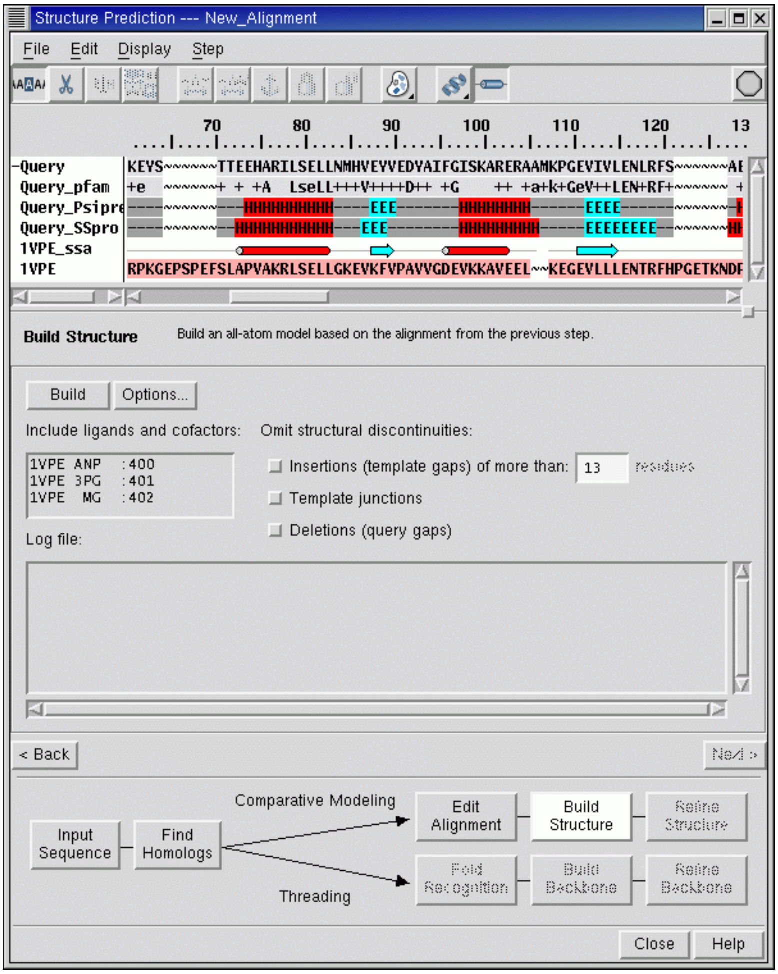
When you click Next at the end of the Edit Alignment step, the Check Alignment warning may appear. The warning lists possible alignment problems that you may want to fix before continuing to the next step. For example, gaps in secondary structure elements of the template, or gaps in the query that are aligned to secondary structure elements of the template.

If you do not want to make corrections to the alignment, click Continue to go to the next step. To make corrections based on the information in the error message, make a note of the gap locations given, click Cancel to close the message box while remaining in Edit Alignment, and change the alignment accordingly.

# Comparative Modeling: Build Structure

## 6.1 The Build Process

The Build Structure step builds a model structure of the query sequence based on the templates selected in the previous step. Figure 6.1 shows the initial view of the Build Structure step.



```
              70        80        90        100       110       120       13
      ....|....|....|....|....|....|....|....|....|....|....|....|....|....
-Query        KEYS~~~~~~~TTEEHARILSELLNMHVEYVEDYAIFGISKARERAAMKPGEVIVLENLRFS~~~~~~~AE
 Query_pfam   +e  ~~~~~~~+ + +A  LseLL+++V++++D++ +G    ++ +a+k+GeV++LEN+RF+~~~~~~~ +
 Query_Psipre ----~~~~~~~---HHHHHHHHHH----EEE------HHHHHHHH------EEEE------~~~~~~~-H
 Query_SSpro  ----~~~~~~~--HHHHHHHHHHH---EEE-------HHHHHHHHH-----EEEEEEEE--~~~~~~~HH
 1VPE_ssa
 1VPE         RPKGEPSPEFSLAPVAKRLSELLGKEVKFVPAVVGDEVKKAVEEL~~KEGEVLLLENTRFHPGETKNDF
```

Build Structure  Build an all-atom model based on the alignment from the previous step.

Include ligands and cofactors:

```
1VPE ANP  :400
1VPE 3PG  :401
1VPE  MG  :402
```

Omit structural discontinuities:

- Insertions (template gaps) of more than: 13 residues
- Template junctions
- Deletions (query gaps)

Log file:

*Figure 6.1. The* Build Structure *step, initial view.*

The Build process includes the following steps:

1. Coordination of the copying of backbone atoms for aligned regions and side chains of conserved residues
2. Building insertions and closing deletions in the alignment

The Build process may also include the following steps:

- Side-chain prediction of non-conserved residues
- Minimization of all atoms not derived directly from the template

You can specify which optional steps to perform by clicking Options and making selections in the dialog box. See Section 6.2.3.

## 6.2 Preparing to Build

If you want to build a model using multiple templates or including ligands or cofactors, you must specify how these are to be treated before clicking the Build button to start the building process.

### 6.2.1 Multiple Templates (Set Template Regions)

If you intend to use multiple templates to build a model of the query, you will have already selected them and ensured that they are structurally aligned (i.e., rotated into a common coordinate frame; see Section 4.2.5). In the Build Structure step, before starting the structure build by clicking the Build button, you must specify which template to use for each region of the query. If you do not select regions from other templates, the topmost template in the table will be used for the entire query.

To enter the mode that allows you to set the template regions, click the Set Template Regions toolbar button:

Highlight the region of each template in the sequence viewer that you want to use. Click the Set Template Regions button again to end the selection. (You may need to scroll over and select again if the desired region is wider than the display window.)

The Set Template Regions tool helps make sure that only one template residue is assigned to a query residue at one time. Regions of a template not being used to model the query are colored dark gray (regardless of coloring schemes). When you begin to set template regions, only the first template is in use. As you select regions in the second template, the tool automatically deselects the corresponding residues in the first template. If you select regions for use from a third template, the corresponding regions of the first and second templates are deselected, and so on.

### 6.2.2 Including Ligands and Cofactors

Any cofactors and ligands present in the template you have chosen (except HOH) are listed in the Include ligands and cofactors box. Click in the list to select one or more ligands and cofactors. Your selections are highlighted in yellow in the Workspace.

### 6.2.3 Build Options

Click Options to open the Build Structure–Options dialog box and specify which optional model building steps to perform:

- Retain [side-chain] rotamers for conserved residues
- Optimize side chains

  If you select Optimize side chains, the third option becomes available:

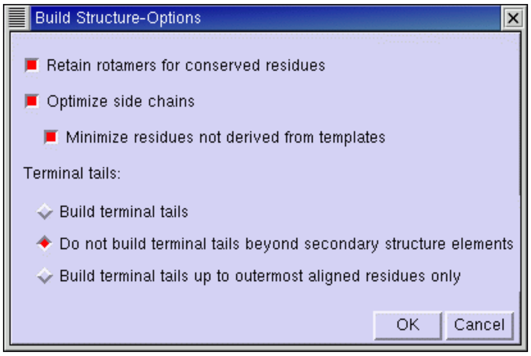
  - Minimize residues not derived from templates

By default, all three of the above options are selected.

The following options are available for the treatment of terminal tails:

- Build terminal tails
- Do not build terminal tails beyond secondary structure elements
- Build terminal tails up to outermost aligned residues only

By default, the second of these options is selected.



[✓ Retain rotamers for conserved residues]

[✓ Optimize side chains]

[✓ Minimize residues not derived from templates]

Terminal tails:

Build terminal tails

[✓ Do not build terminal tails beyond secondary structure elements]

Build terminal tails up to outermost aligned residues only

***Figure 6.2. The*** Build Structure–Options ***dialog box.***

### 6.2.4 Omitting Structural Discontinuities

You can disable the building of structural discontinuities arising from insertions, template junctions, and deletions. The discontinuities are capped with NMA and ACE, so the final structure is discontinuous. If you choose any of these options, you should check that your final structure is reasonable.

If there is an insertion in the query (a template gap) that requires the building of a long loop, you can instead cut the query sequence and cap it with NMA and ACE, and not build the loop, by selecting the Insertions (template gaps) option under Omit structural discontinuities, and enter the maximum length of loops that will be built in the text box. This option could be useful where there are long insertions that are not in the region of interest.

Likewise, if there are deletions in the query (a query gap), or if there is a junction between two templates, you can choose to cap the ends of the two pieces and not try to join them in building the structure. To do this, select Deletions (query gaps) or Template junctions under Omit structural discontinuities.

## 6.3 Running the Build Structure Job

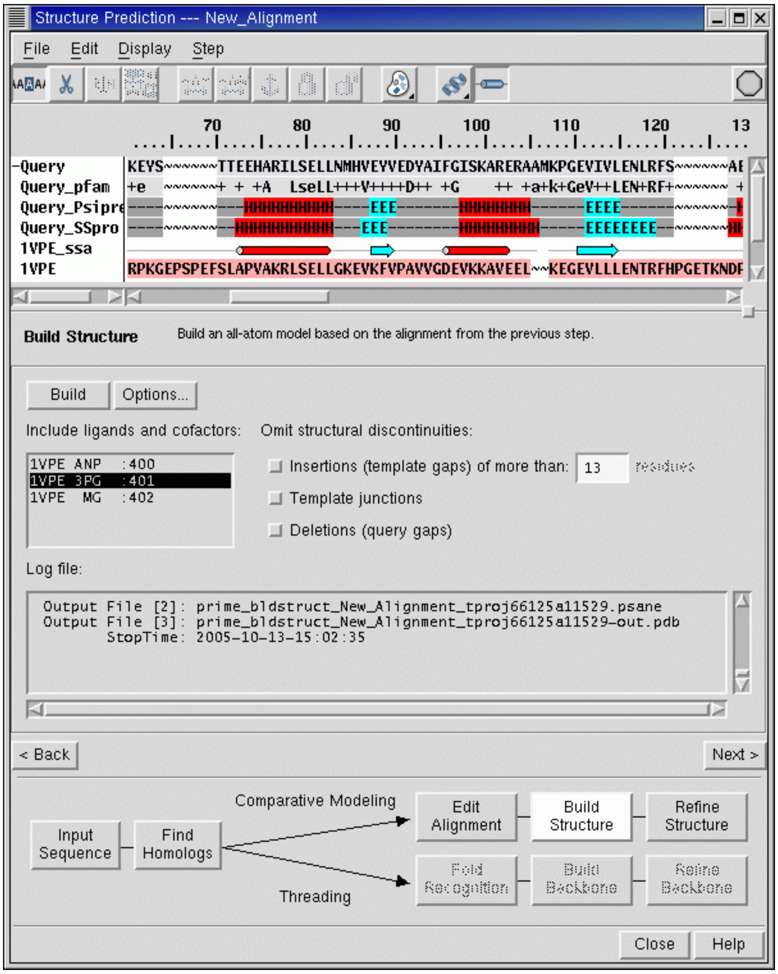
When you click Build, the model-building program begins, and its log is displayed in real time in the Log file window (see Figure 6.3). It can also be viewed in the Job Monitor (click the octagon in the upper right corner of any Prime–SP panel to open the Monitor panel). Finally, the build log is saved to a log file in the directory where Prime is running.

First, the log lists the chosen templates. If you did not intend to choose more than one template, or if you neglected to align the templates, stop the job and restart it with the correct template. Jobs can be stopped, paused, and resumed from the Monitor panel. See Section 3.5 on page 40 for more information on Job Control.

The next part of the log file gives information about the build process for template transition regions, then for insertions, and then for side chains, followed by non-template regions. The last line of the log describes the diagnosis and success of model building.

### 6.3.1 Missing Parameters

When the Build job begins, it checks for missing parameters and reports them in its log. If you see a warning in the Build log, you may need to create new parameters for the ligand or cofactor. This can be done using the command-line utility `hetgrp_ffgen`. The procedure for generating missing parameters is described in Section 14.3.



```
-Query       KEYS~~~~~~TTEEHARILSELLNMHVEYVEDYAIFGISKARERAAMKPGEVIVLENLRFS~~~~~~AE
 Query_pfam  +e   ~~~~~~+ + +A  LseLL+++V++++D++ +G    ++ +a+k+GeV++LEN+RF+~~~~~~ +
 Query_Psipre----~~~~~~---HHHHHHHHHH----EEE-------HHHHHHHH------EEEE------~~~~~~-H
 Query_SSpro ----~~~~~~--HHHHHHHHHHH---EEE--------HHHHHHHHH-----EEEEEEEE--~~~~~~HH
 1VPE_ssa
 1VPE        RPKGEPSPEFSLAPVAKRLSELLGKEVKFVPAVVGDEVKKAVEEL~~KEGEVLLLENTRFHPGETKNDF
```

Include ligands and cofactors:

```
1VPE ANP  :400
1VPE 3PG  :401
1VPE  MG  :402
```

Omit structural discontinuities:

- ☐ Insertions (template gaps) of more than: 13 residues
- ☐ Template junctions
- ☐ Deletions (query gaps)

Log file:

```
 Output File [2]: prime_bldstruct_New_Alignment_tproj66125a11529.psane
 Output File [3]: prime_bldstruct_New_Alignment_tproj66125a11529-out.pdb
        StopTime: 2005-10-13-15:02:35
```

***Figure 6.3. The*** Build Structure ***panel after building.***

## 6.3.2 Output From Build Structure

Once the model building calculations are complete, the model is displayed in the Workspace superimposed on the template, using the color scheme Atom PDB B Factor (Temperature Factor). Blue atoms are those derived directly from the template; red atoms have been predicted or modeled. To restore this color scheme, choose Atom PDB B Factor (Temperature Factor) from the Color all atoms by scheme toolbar button menu in the main window.

Build Structure also creates an atom set named non_template_residues which can be found in the Sets folder of the Atom Selection dialog box. This set of inserted residues is generally chosen to undergo refinement.

The sequence viewer window contains the query sequence, the templates, and the sequence of the newly predicted structure in that order. To view only the model structure, select View Structure from the Display menu and then select Predicted Only. (The other options are All and None.)

## 6.4 Technical Notes for the Build Structure Step

The Build Structure step uses the following resources and methods:

- The OPLS2000 all-atom force field (protein optimized) for energy scoring of proteins; the OPLS2001 force field for ligands and other non-amino-acid residues.
- A Surface Generalized Born (SGB) continuum solvation model for treating solvation energies and effects.
- Residue-specific side-chain rotamer and backbone dihedral libraries, derived from the non-redundant data sets extracted from the PDB, and representing values most commonly observed in protein crystal structures. The libraries allow for both the evaluation of existing rotamer/dihedral values and the systematic prediction of new ones.

Both Build Structure and the step that follows, Refine Structure, can treat the 20 standard amino acids as well as modified residues and ligands. Parameter files exist for most modified residues and several common ligands. Additional parameter files can be generated as described in Section C.4 on page 179.

Model building uses the following procedure:

1. Non-conserved residues are mutated to the desired identity. Side chains are added by finding the first rotamer in the library that does not produce a clash.
2. Insertions, deletions, and template transitions are built. These are cases in which the backbone itself needs to be reconstructed, either due to gaps in the alignment or template transitions that produce gaps in the 3D structure. These gaps are closed by reconstruction of the affected region ab initio, using a backbone dihedral library. If no reasonable conformation can be generated that closes the loop structure, the region being reconstructed will be expanded until closure is possible.
3. Gap reconstruction is done by finding any single loop conformation that closes the structure and is physically reasonable. If multiple conformations are found, the one that deviates from the template structure as little as possible is chosen. No attempt is made to

perform an exhaustive optimization of the region.

4. Side-chain prediction (optional). Side chains may be optimized. The two choices are either: (a) optimize all side chains from scratch, or (b) optimize only those that are not in the template.
5. Minimization of non-template regions (optional). Any portions of the backbone that were not derived directly from one of the templates (and were thus reconstructed ab initio) are subject to a local minimization as described in Section 7.4 on page 72.

# Comparative Modeling: Refine Structure

## 7.1 Overview of the Refine Structure Step

A model structure built in the Build Structure step may require further refinement. If there are insertions or deletions in the alignment between query and template, it is recommended that you perform a loop refinement, since insertions and deletions are most frequently found in loop regions. A loop refinement will enumerate and score multiple loop conformations constructed following an ab initio procedure. Scoring involves side-chain predictions and all-atom minimizations.

A side-chain prediction or minimization of a region you specify (a set of residues, for example) can also be performed in this step. Select the desired procedure from the Task options, and specify a region or regions. The Run button remains dimmed until you have selected some part (or the whole) of the structure for refinement. For the Refine loops task, you select loops for refinement by clicking the check box in the Run column. For Predict side chains and Minimize, the standard Maestro atom selection options are offered, including the blue Select button that opens the Atom Selection dialog box. (See Section 2.6 on page 23, the online help, or Chapter 5 of the *Maestro User Manual* for more information about atom selection.)

In the Sets folder of the Atom Selection dialog box, there may be automatically generated sets such as `non_template_residues`. If the Protein Report found unusual values for backbone dihedrals or side-chain dihedrals, the set *name*`_pred_warn_backbone_dihedrals` or *name*`_pred_warn_sidechain_dihedrals` will be present. All these atom sets are good choices for refinement.

When you have specified a region for refinement, the Run button becomes available.

It may be helpful to view a structure's Protein Report before (and after) choosing a refinement task and selecting a region for refinement. The Protein Report provides information on certain properties of the structure, noting unusual values that may reflect structure problems. Select the structure to be refined from the Structures table and click the View Report button to open the Protein Reporting Panel. A Protein Report is created for each built structure, and a new one is generated for each refined structure. For more on the Protein Reporting Panel, see Section 7.6 on page 78.

If you would like to modify the model produced by Build Structure before doing refinement (for example, if you want to add a water molecule to the binding site):

1. Click Add to Project Table and hide the Structure Prediction panel.
2. In Maestro, make a composite protein-water entry.
3. With this entry included in the Workspace, open the Prime–Refinement panel, from which you can run the same refinement jobs as in Refine Structure.

When refinement procedures are finished, the Comparative Modeling run can be completed by adding the refined structure to the Project Table.

## 7.2 Loop Refinement

By default, the Refine loops task is selected when you enter the Refine Structure step. The Structures table on the right indicates the selected structure with a yellow highlight. The Loops table on the left lists all the loop features of length three or greater in the selected structure, as shown in Figure 7.1. (Loops of one or two residues are not listed, as they are unsuitable for the refinement protocol.)

For each loop, the table includes a Run check box, a feature name such as loop1, and the numbers of the first (Res1) and last (Res2) residues in the loop. The first loop feature in the table is selected (highlighted in yellow) by default, but no feature is designated to be run (no check box in the Run column is selected), and therefore the Run button is dimmed.

**To refine loops:**

1. Select a loop by clicking its name.

   The loop row is highlighted in yellow. You can change settings for the selected loop, such as Maximum CA movement, in the Refine Structure-Options dialog box—see Section 7.5.

2. To designate this loop for refinement, select the check box in the Run column.

   The Run button becomes available.

3. If necessary, shorten the loop region to be refined by changing the first and last residue numbers (Res1 and Res2).

   Longer loops require more time to run. For default sampling, the time required scales approximately linearly with the length of the loop. It is recommended that extended sampling be used for loops longer than five residues. See Section 7.5.

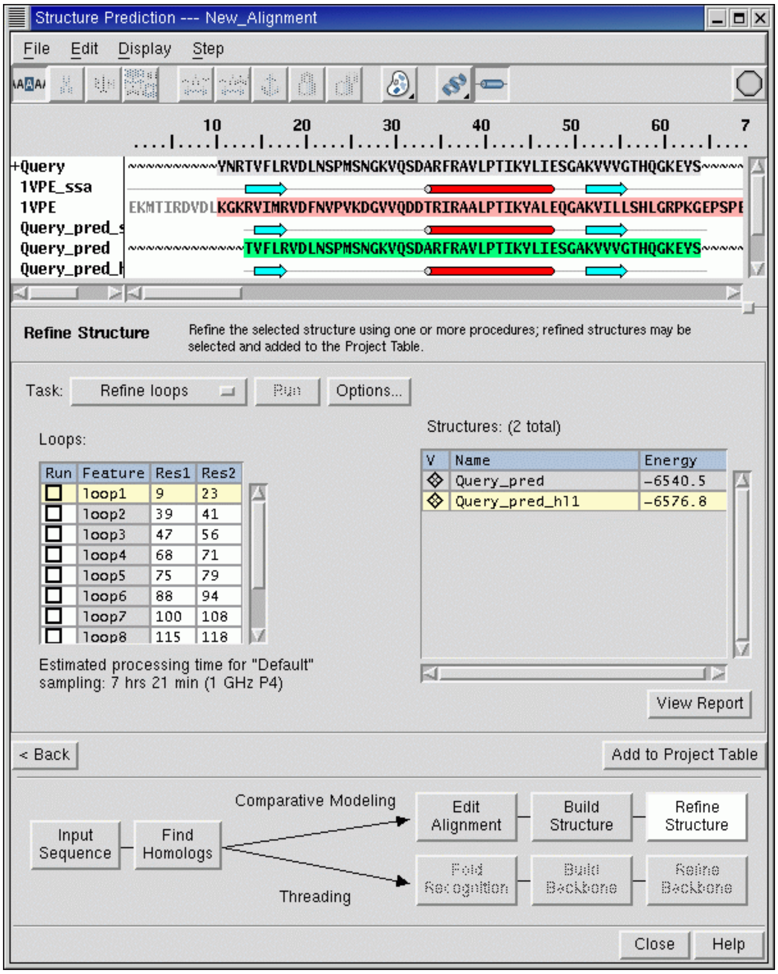
4. (Optional) Click Options to select a sampling method and make settings for this loop in the Refine Structure-Options dialog box.

   If the loop is longer than five residues, you should select a sampling method other than the default. If you select Ultra Extended sampling, you can only refine one loop.

5. Repeat the procedure from Step 1 for as many loops as you want to refine.
6. Click Run to start the loop refinement job.

If several loops are selected, they are refined in series. The first one selected is refined, the best scoring structure for that loop is determined, that structure is used in refining the next loop, and so on. When all loops have been refined, one final structure is returned. If only one loop is refined, the four highest-scoring structures are returned by default.

You can also select two loops for cooperative refinement. In this procedure refinement of the two loops is iterated until there is no further structural change. For more information on the cooperative loop refinement process, see Section 7.8.2 on page 81.



```
+Query          ~~~~~~~~~~YNRTVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGKEYS~~~~~
 1VPE_ssa
 1VPE           EKMTIRDVDLKGKRVIMRVDFNVPVKDGVVQDDTRIRAALPTIKYALEQGAKVILLSHLGRPKGEPSPE
 Query_pred_s
 Query_pred     ~~~~~~~~~~~~~TVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGKEYS~~~~~
 Query_pred_h
```

Refine the selected structure using one or more procedures; refined structures may be selected and added to the Project Table.

| Run | Feature | Res1 | Res2 |
|---|---|---|---|
| ☐ | loop1 | 9 | 23 |
| ☐ | loop2 | 39 | 41 |
| ☐ | loop3 | 47 | 56 |
| ☐ | loop4 | 68 | 71 |
| ☐ | loop5 | 75 | 79 |
| ☐ | loop6 | 88 | 94 |
| ☐ | loop7 | 100 | 108 |
| ☐ | loop8 | 115 | 118 |

Estimated processing time for "Default" sampling: 7 hrs 21 min (1 GHz P4)

| V | Name | Energy |
|---|---|---|
| ◈ | Query_pred | -6540.5 |
| ◈ | Query_pred_h11 | -6576.8 |

***Figure 7.1. The*** Loops ***task of the*** Refine Structure ***step.***

When a loop refinement begins, a validation program checks the loop features of the structure. Apparent errors are reported in a warning dialog box. You may choose to Run All Features or to Run Only Valid Features, but it is recommended that you make a note of the invalid features, click Cancel, and if appropriate correct the structure.

## 7.3 Side Chain Prediction

By default, side-chain prediction returns a single structure prediction.

**To request multiple side-chain predictions:**

1. Click Options.
2. Under Side-Chain Prediction, enter the desired number in the Number of structures to return text box.
3. Click OK to close the Refine Structure-Options dialog box.

**To predict side-chain conformations:**

1. Select Predict side chains from the Task menu.

   The panel displays atom selection options under the heading Residues for side chain refinement.

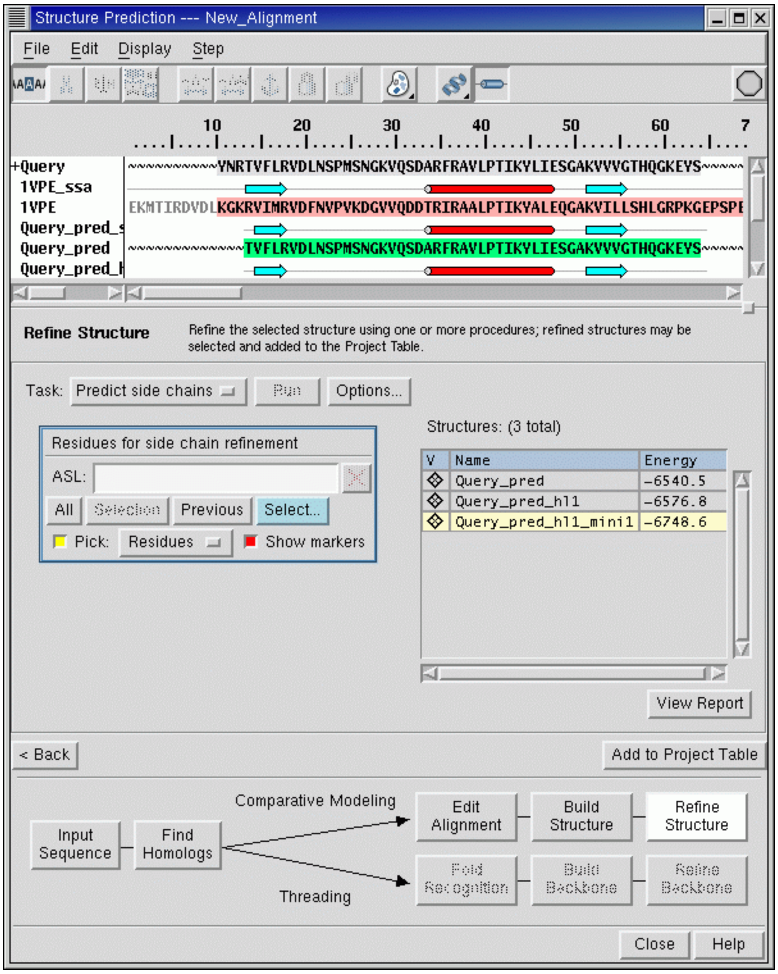
2. Specify residues for side-chain refinement using the picking controls.

   For example, the Atom Selection dialog box provides proximity options that can be used to select all residues within a certain distance of a ligand. It is not usually necessary to predict side chains on all residues, but if you want to do so, click All. Atom selection and the Atom Selection dialog box are briefly described in Section 2.6 on page 23. For more information, see the online help or Chapter 5 of the *Maestro User Manual*.

3. Click Run to start the side-chain prediction job.

## 7.4 Energy Minimization

The Minimize refinement task performs a truncated-Newton energy minimization, using the OPLS2000 all-atom force field (protein-optimized) for proteins and OPLS2001 for ligands and cofactors, and treating solvation energies and effects via the Surface Generalized Born (SGB) continuum solvation model.



```
+Query        ~~~~~~~~~~YNRTVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGKEYS~~~~~
 1VPE_ssa
 1VPE         EKMTIRDVDLKGKRVIMRVDFNVPVKDGVVQDDTRIRAALPTIKYALEQGAKVILLSHLGRPKGEPSPE
 Query_pred_s
 Query_pred   ~~~~~~~~~~~~~TVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGKEYS~~~~~
 Query_pred_h
```

Refine the selected structure using one or more procedures; refined structures may be selected and added to the Project Table.

Task: Predict side chains

Residues for side chain refinement

ASL:

Structures: (3 total)

| V | Name | Energy |
|---|---|---|
| ◈ | Query_pred | -6540.5 |
| ◈ | Query_pred_h11 | -6576.8 |
| ◈ | Query_pred_h11_mini1 | -6748.6 |

***Figure 7.2. The*** Side Chains ***task of the*** Refine Structure ***step.***

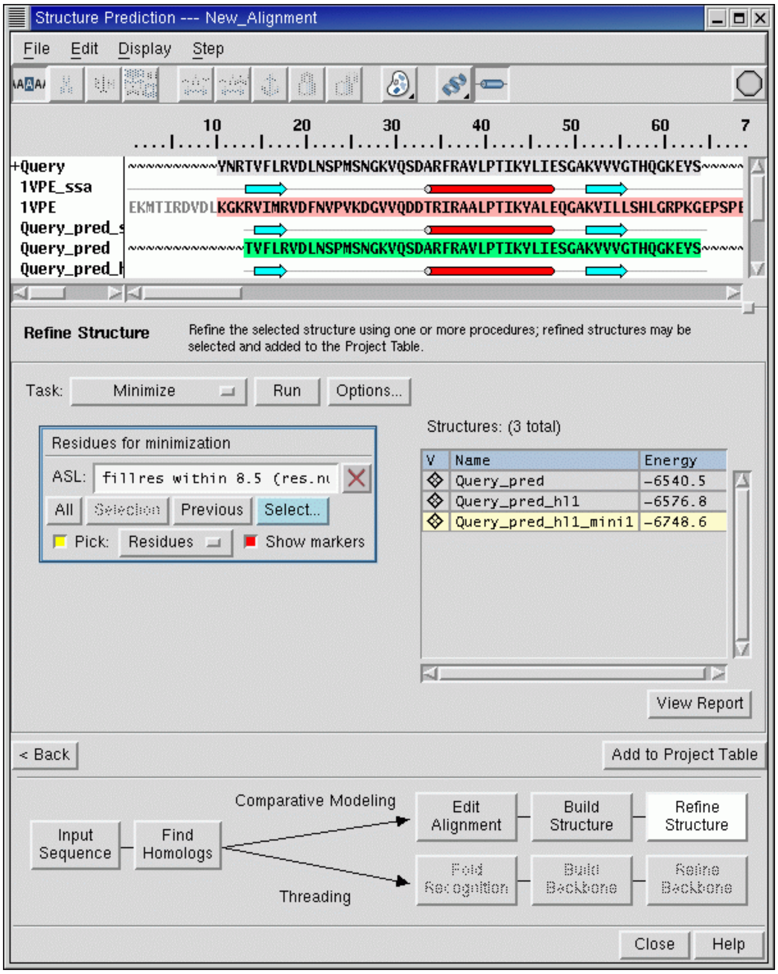
**To minimize all or part of the structure selected:**

1. Select Minimize from the Task menu.

   Atom selection controls are displayed under the heading Residues for minimization.

2. Use the picking controls to specify a region for minimization.

   It is recommended that you select a region or a subset of residues for minimization rather than performing an all-atom minimization of the entire structure. You can check the Sets folder of the Atom Selection dialog box for useful generated sets such as non_template_residues and (if applicable) *name*_pred_warn_backbone_dihedrals and



```
+Query        ~~~~~~~~~~~YNRTVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGKEYS~~~~~
 1VPE_ssa
 1VPE         EKMTIRDVDLKGKRVIMRVDFNVPVKDGVVQDDTRIRAALPTIKYALEQGAKVILLSHLGRPKGEPSPE
 Query_pred_s
 Query_pred   ~~~~~~~~~~~~~~TVFLRVDLNSPMSNGKVQSDARFRAVLPTIKYLIESGAKVVVGTHQGKEYS~~~~~
 Query_pred_h
```

**Refine Structure** Refine the selected structure using one or more procedures; refined structures may be selected and added to the Project Table.

ASL: fillres within 8.5 (res.nu

Structures: (3 total)

| V | Name | Energy |
|---|---|---|
| ◈ | Query_pred | -6540.5 |
| ◈ | Query_pred_h11 | -6576.8 |
| ◈ | Query_pred_h11_mini1 | -6748.6 |

***Figure 7.3. The*** Minimize ***task of the*** Refine Structure ***step.***

*name*_pred_warn_sidechain_dihedrals. For more information on using the picking controls, see Section 2.6 on page 23, the online help, or Chapter 5 of the *Maestro User Manual*.

3. Click Run to start the minimization job.

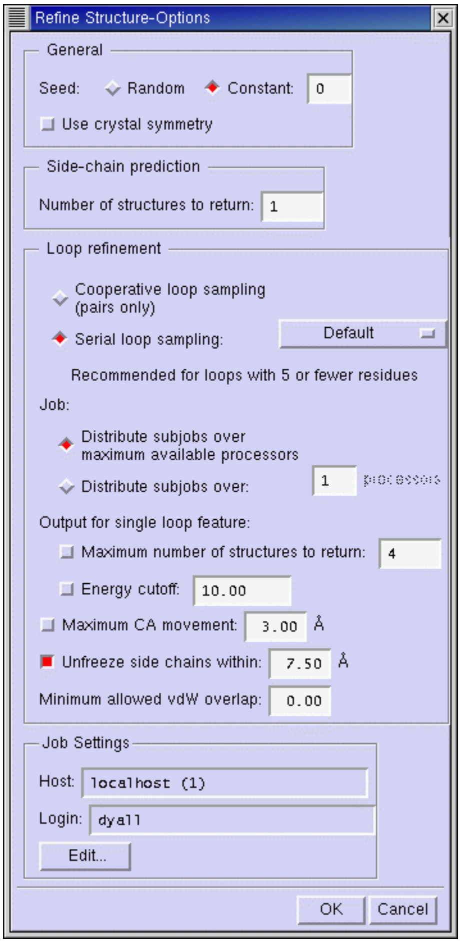
## 7.5 Structure Refinement Options

The Refine Structure–Options dialog box for the Prime–SP Refine Structure step offers General options, Side-Chain Prediction options, Loop Refinement options, and Job Settings options (see Figure 7.4).

To open this dialog box, click Options.

### 7.5.1 General Options

The General section supplies options that apply to all types of refinement:


Refine Structure-Options

General

Seed: Random [✓ Constant:] 0

☐ Use crystal symmetry

Side-chain prediction

Number of structures to return: 1

Loop refinement

Cooperative loop sampling (pairs only)

[✓ Serial loop sampling:] Default

Recommended for loops with 5 or fewer residues

Job:

[✓ Distribute subjobs over maximum available processors]

Distribute subjobs over: 1 processors

Output for single loop feature:

☐ Maximum number of structures to return: 4

☐ Energy cutoff: 10.00

☐ Maximum CA movement: 3.00 Å

[✓ Unfreeze side chains within:] 7.50 Å

Minimum allowed vdW overlap: 0.00

Job Settings

Host: localhost (1)

Login: dyall

Edit...

OK Cancel

**Figure 7.4.** ***The*** Refine Structure–Options ***dialog box with*** Default ***sampling selected.***

Seed

Specify the seed for the random-number generator used in side-chain predictions. The seed is used for side-chain prediction in both the loop refinement and the side-chain prediction tasks.

- Random: use a random seed.
- Constant: use the integer given in the text box as the seed. The default is 0.

Use crystal symmetry

If crystal symmetry is known for this protein, apply periodic boundary conditions so that the crystal symmetry is satisfied.

### 7.5.2 Side-Chain Prediction Options

In this section you can specify the number of structures to return from side-chain prediction. The default is 1. To return multiple structures, enter the desired number in the Number of structures to return text box.

### 7.5.3 Loop Refinement Options

The Loop Refinement section provides options for selecting the sampling method, job options, output options, and some general options.

Prime can perform cooperative loop sampling as well as serial loop sampling. Both the size of the loop and the sampling options chosen affect the time needed for loop refinement. The options are:

Cooperative loop sampling

Select this option to refine two loops together. Each loop is refined in turn until the refinement has converged for both loops. You must have only two loops selected to use this option.

Serial loop sampling

Select this option to refine each loop independently. The option menu provides five levels of sampling accuracy: Default, Extended Low, Extended Medium, Extended High, Ultra Extended. The text below the option and menu displays the recommended loop length for the selected accuracy.

Extended sampling samples loop conformations more thoroughly, using a series of loop constraint settings. Up to eight processors can be used simultaneously to perform loop refinement. Extended sampling can be run at four different sampling levels: Low, Medium, High, and Ultra.

### Output for single loop feature

These options determine how many structures are returned:

- Maximum number of structures to return: Select this option to return more than one structure for each loop specified for refinement. The default when this option is selected is to return 4 structures for each loop.
- Energy cutoff: Select this option to prevent higher-energy structures from being returned. Structures whose energy is higher than that of the lowest-energy structure by more than this amount are not returned. The default cutoff when this option is selected is 10 kcal/mol.

### Job

These options allow you to set a limit on the the number of processors over which to distribute the loop sampling subjobs. Each sampling method runs in several stages, and each stage can use a different number of processors. The maximum number of processors that can be used by the different sampling methods are:

| | |
|---|---|
| Default | 1 |
| Extended Low | 2 |
| Extended Medium | 4 |
| Extended High | 8 |
| Ultra Extended | 8*L (L is the loop length) |

- To set the maximum number of processors, select Distribute subjobs over and enter a value in the text box.
- To make use of as many processors as are available, select Distribute subjobs over maximum available processors.

The following settings are applied to the loops you have selected for refinement.

- Maximum CA movement: To limit the extent to which alpha carbon atoms can move from their original positions, select this option, and enter a limit in the text box. The default distance is 3.0 Å. The extended sampling procedure tests multiple values for Maximum CA movement, so this option is unavailable when extended sampling is selected.
- Unfreeze side chains within: Select this option to allow side chains near the specified loop to move, and specify a distance in the text box. The default distance is 7.5 Å.
- Minimum allowed vdW overlap: By default, this value is set to 0.70, meaning that the distance between atoms must be at least 70% of their ’ideal’ van der Waals separation. The extended sampling procedure tests multiple values for the minimum overlap, and therefore this option is unavailable when extended sampling has been selected.

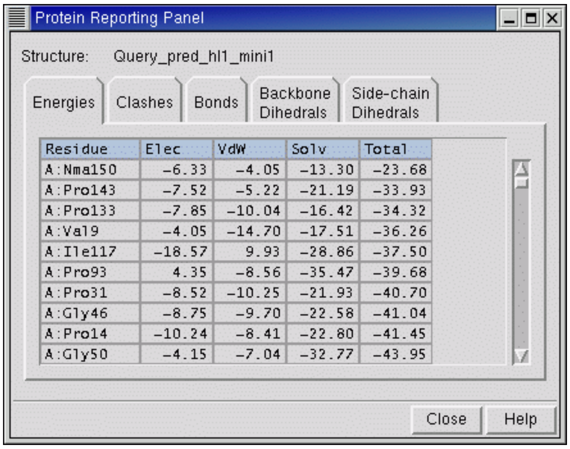
### 7.5.4 Job Settings Options

The Job Settings section lists the Host and the Login to use for jobs run from the Refine Structure step. Click the Edit button to open the Structure Prediction–Job Options panel (shown in Figure 3.2) if you want to change the host and/or login. This panel can be opened in any step by selecting Job Options from the Prime File menu.

## 7.6 The Protein Reporting Panel

For each protein structure model built or refined, a Protein Report is created. The information in this report may suggest regions of the model that should be investigated and perhaps selected for additional refinement.

Select a structure and click the View Report button to open the Protein Reporting Panel (Figure 7.5). This panel reports structure properties and marks unusual values that may suggest particular problems with the structure. Click on a tab to open a table showing unusual values or violations of that property. The column widths in these tables can be adjusted by dragging the column border with the middle mouse button, though the table cannot be re-sorted. If you click on a residue in any of the tables, it is highlighted in the Workspace. For most properties, a folder is not displayed if no unusual values are found. The contents of the tables are described in Table 7.1.



| Residue | Elec | VdW | Solv | Total |
|---|---|---|---|---|
| A:Nma150 | -6.33 | -4.05 | -13.30 | -23.68 |
| A:Pro143 | -7.52 | -5.22 | -21.19 | -33.93 |
| A:Pro133 | -7.85 | -10.04 | -16.42 | -34.32 |
| A:Val9 | -4.05 | -14.70 | -17.51 | -36.26 |
| A:Ile117 | -18.57 | 9.93 | -28.86 | -37.50 |
| A:Pro93 | 4.35 | -8.56 | -35.47 | -39.68 |
| A:Pro31 | -8.52 | -10.25 | -21.93 | -40.70 |
| A:Gly46 | -8.75 | -9.70 | -22.58 | -41.04 |
| A:Pro14 | -10.24 | -8.41 | -22.80 | -41.45 |
| A:Gly50 | -4.15 | -7.04 | -32.77 | -43.95 |

*Figure 7.5. The* Protein Reporting Panel*.*

*Table 7.1. Description of folders in the Protein Reporting panel.*

| Folder | Contents |
|---|---|
| Energies | Component (Coulombic, short-range, and solvation) energies and total energy by residue, sorted by decreasing total energy. |
| Clashes | Any pair of atoms that produces a steric clash is reported. |
| Bonds | Any bond lengths that significantly deviate from the ideal bond lengths are reported. |
| Angles | Any angles that significantly deviate from the ideal angles are reported. |
| Backbone Dihedrals | All backbone dihedral angles are reported. Those that significantly deviate from anything in the backbone dihedral library are marked with an asterisk. |
| Side-chain Dihedrals | All side-chain dihedral angles are reported. Those that significantly deviate from anything in the side-chain dihedral library are marked with an asterisk. |
| Missing H-bonds | Any atoms that "should" have H-bonds but do not appear to are reported. Atoms that are highly solvent-accessible are skipped, as they are presumably H-bonded to water. |

## 7.7 Saving and Exporting Refined Structures

When refinement is complete, the refined structures can be appended to the Maestro project by clicking Add to Project Table. You can export any Project Table entry, such as a refined structure, to a file of another format.

**To export a Project Table entry:**

1. In the Project Table, select the structure you want to export, then choose Export from the Table menu.
2. In the Export panel:
   a. Select the format (Maestro, MMOD, PDB, Mol2, SD).
   b. Enter a file name.
   c. Ensure the Structure Source To Be Exported option is set to Selected Entries.
   d. Click the Export button.

## 7.8 Technical Details for the Refine Structure Step

### 7.8.1 Loop Refinement

Loop refinement using the Default option proceeds as follows:

1. The loop is reconstructed using the backbone dihedral library, by building up half from each direction. The resolution starts off very coarsely, becoming finer until it manages to produce a required number of physically realistic loop conformations (based on the length of the loop).
2. The large number of loops generated in the first step are clustered, and representatives of each cluster are selected.
3. These cluster representatives are then scored as follows:
    a. Side chains are re-added to the loop residues, as well as any residues within a specified cutoff, according to the side-chain optimization procedure.
    b. The loop and contact side chains are minimized.
    c. The energy is calculated.

4. The best scoring loop structures are returned.

Loop prediction with extended sampling is specifically designed to overcome sampling problems with long loops, in which the number of residues results in an unwieldy number of possible loop conformations.

Extended sampling uses the following procedure:

1. Two initial predictions are run on the loop. These are intended to coarsely sample conformational space and are carried out with two slightly different sets of prediction parameters. Each initial prediction returns up to four top-scoring conformations, resulting in up to eight structures. These eight structures are intended as initial points for finer sampling of what appears to be favorable regions (basins) of conformational space.
2. Each of the eight structures generated in Step 1 is run through another stage of refinement, with a 5 Å restriction on Cα movement. This is intended to sample the basins more finely.
3. The eight refined structures from Step 2 are combined with the original structures from Step 1, and this set of 16 structures is ranked by energy. The four top-scoring structures are then subjected to a final stage of refinement, with a 2 Å restriction on Cα movement. This is intended to perform fine-grained sampling of the best basins.

This procedure defines the “Extended High” sampling procedure. The “Extended Medium” and “Extended Low” carry four and two structures, respectively, into Step 2 and Step 3.

In ultra-extended sampling, the three steps of extended sampling are followed by the steps below.

4. The top four structures are selected, and two predictions run on each, with part of the loop frozen. Runs are done with one residue held fixed at each end, two residues held fixed at each end, and so on, up to half the total length.
5. Step 2 is repeated with the top eight structures cumulatively generated, then Step 3 is repeated with the top eight structures cumulatively generated.

### 7.8.2 Cooperative Loop Refinement

Cooperative loop refinement proceeds as follows:

1. A list of side chains that will be optimized is generated. By default this list includes any residue with an atom within 7.5 Å of either loop 1 or loop 2.
2. Loop 1 is sampled, allowing side chains in the list from Step 1 to move, holding the backbone of loop 2 fixed at its original conformation. Eight loop conformations are generated.
3. Loop 2 is sampled, allowing side chains in the list from Step 1 to move, holding the backbone of loop 1 fixed at its original conformation. Eight loop conformations are generated. This step runs concurrently with Step 2.
4. Conformations of loop 1 from Step 2 with loop 2 from Step 3 are merged into a set of structures.
5. All side chains in the list determined in Step 1 are reoptimized, keeping only those structures whose energy is below the energy cutoff.
6. All residues in both loops are then minimized, inclucing the backbone, keeping only those structures whose energy is below the energy cutoff.
7. The 4 best new structures are taken as input for resampling, starting from Step 2. The process is repeated until convergence is obtained.

### 7.8.3 Side-Chain Prediction

The structure refinement program uses the following procedure to re-predict conformations for the side chain set that you select.

1. Side-chain rotamers are randomized for nonconserved residues (the default) or for all residues.

2. Beginning with the first residue to be predicted, the side-chain rotamer library is used to find the rotamer with the lowest energy while keeping all other side chains fixed. Once the process is complete for the first residue, the next residue is treated, and so forth until all have been done once (a single pass has been completed).
3. Once the pass is complete, Step 2 is repeated from the beginning, and then repeated again until the side-chain rotamers appear to be converged (no more changes are occurring).
4. Minimization is run on all of the side-chain atoms (but not backbone atoms) of the residues being treated.

## 7.9 Refinement with Nonstandard Residues

If you want to refine a structure with nonstandard residues, you can make changes to the residues in the structures before refinement. Several nonstandard residues are supported in the Build panel: the neutralized forms of ASP, LYS, ARG, and GLU, which are named ASH, LYN, ARN, and GLH; and the tautomers of HIS (HID=HIS and HIE) and its protonated form, HIP.

Ionization (deprotonation) of four other residues are supported, CYS, SER, THR, and TYR. The names of the ionized residues are CYT (thiolate), SRO (alkoxide), THO (alkoxide), and TYO (phenoxide). In addition, the disulfide-bridged CYS is supported as CYX. To change one of these residues, you can either change the name, or change the structure. For example, you can generate an ionized SER by either of the following methods:

- Changing the residue name to SRO. The HG is deleted and a formal negative charge is added to the OG.
- Deleting the HG (if present) and adding a formal charge to the OG. The residue is renamed to SRO.

# Threading: Fold Recognition

## 8.1 The Threading Path

When sequence searches in the Find Homologs step of Prime–SP fail to find templates, or query-template sequence identity is low, structure prediction may be continued using the Threading path. The Threading path is a series of three Prime–SP steps designed to produce model backbone structures of the query sequence. It uses secondary structure matching and profiles generated from multiple sequence and multiple structure alignments.

The structure prediction process in the Threading path includes the following steps:

1. Fold Recognition is used to find candidate "seed" templates.
2. Composite templates are generated in Build Backbone using multiple structures from seed template families. Low-resolution backbone models are built via alignment of the query to composite templates.
3. Selected backbone models are refined in Refine Backbone.

## 8.2 Overview of the Fold Recognition Step

The Fold Recognition step is designed to find structural templates that could not be found by a sequence search. The use of secondary structure matching is important in finding remote (<15%) homologs. Fold Recognition may identify a family or fold type for an unknown sequence.

The Fold Recognition process begins with the production of a secondary structure prediction profile for the query sequence. Once the query's SSP profile is generated, it can be used to identify candidate templates of known structure. Preliminary models of the alignment of the query to potential templates are ranked and displayed in the Templates table. The best of these preliminary models are selected as "seed" templates and brought into the Build Backbone step.

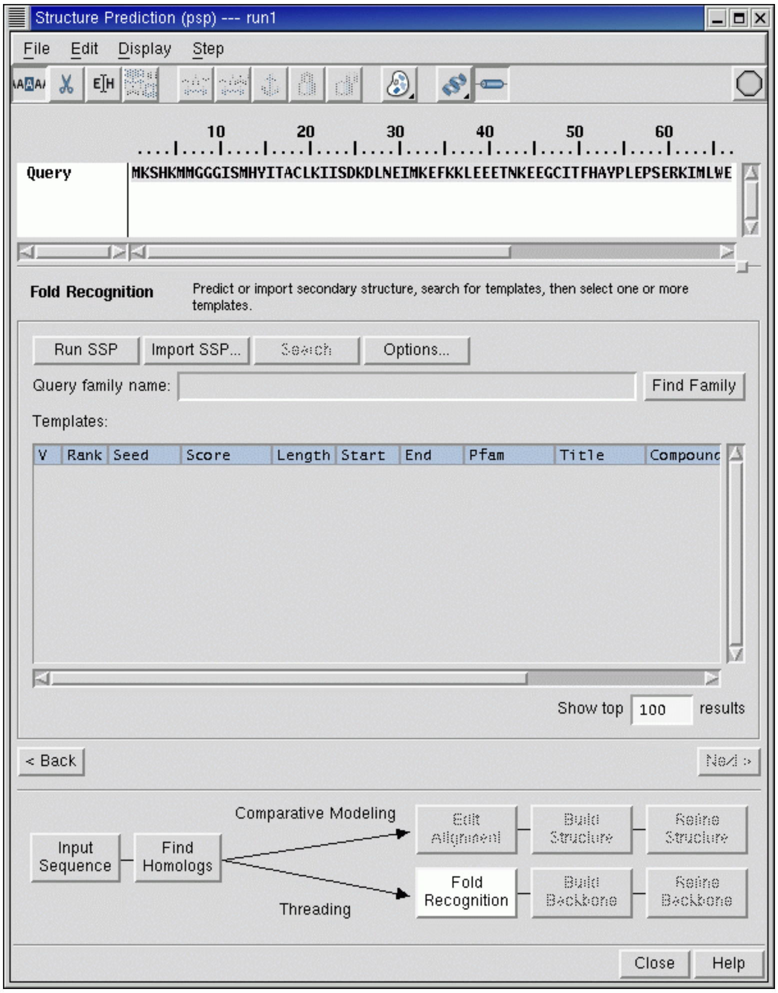
## 8.3 Producing an SSP Profile

Secondary structure predictions to be used in the query's SSP profile may either be imported from a file or generated within Prime. The SSPs can then be exported, deleted, or edited to produce a satisfactory SSP profile for the query.

## 8.3.1 Importing and Exporting SSPs

To import a secondary structure prediction for the query SSP profile, click the Import SSP button and select an SSP file. Files may be FASTA or `m2io` (native Schrödinger) format. If necessary, use the `seqconvert` utility to change the file format. (See Section 14.4 for more information on `seqconvert`.) If the file contains bad data or the sequence doesn't match the query sequence, an error message is displayed.

To export an SSP, right-click on the SSP in the sequence viewer and select Export.



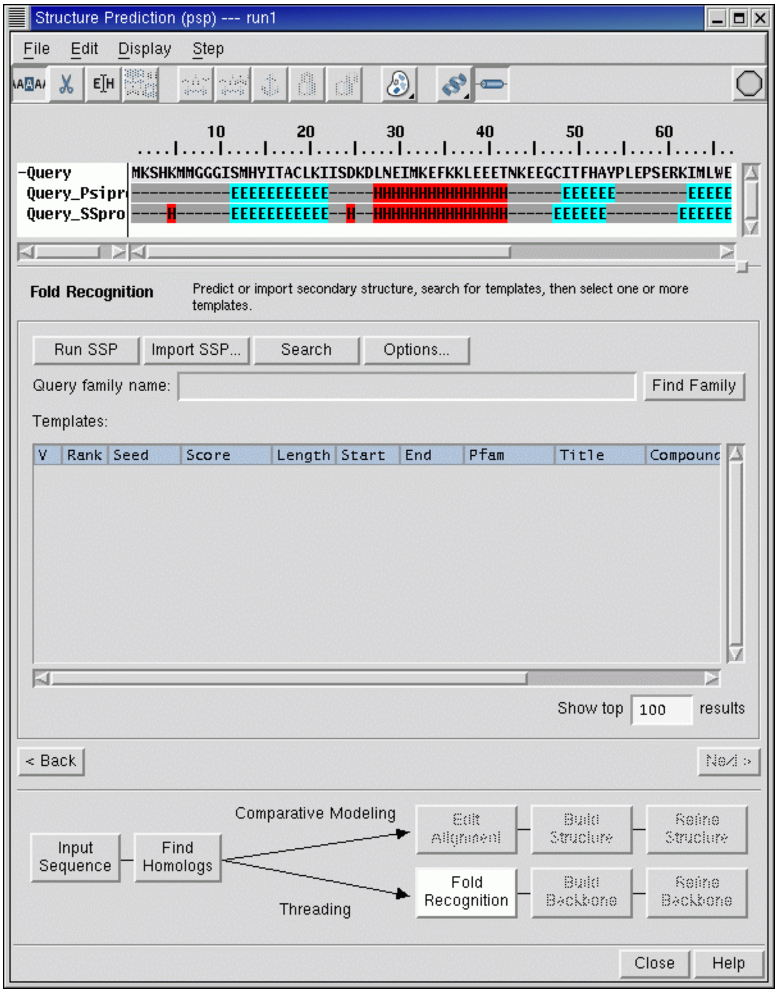
Predict or import secondary structure, search for templates, then select one or more templates.

*Figure 8.1. The* Fold Recognition *panel, initial view.*

## 8.3.2 Generating SSPs

To generate SSPs for the SSP profile, run all available SSP programs by clicking the Run SSP button. One secondary structure prediction program (SSpro) is bundled with Prime. However, two optional third party programs are recommended for optimal results. See Chapter 15 for more information on these programs.

When the secondary structure prediction jobs finish, the SSPs are displayed in the sequence viewer below (as children of) the query sequence. (If an error occurs when running a structure prediction program, an error message is printed to the log file displayed in the Monitor panel.)



```
                   10         20         30         40         50         60
-Query          MKSHKMMGGGISMHYITACLKIISDKDLNEIMKEFKKLEEETNKEEGCITFHAYPLEPSERKIMLWE
 Query_Psipr    -----------EEEEEEEEEEE-----HHHHHHHHHHHHHHH------EEEEEE--------EEEEE
 Query_SSpro    ----H------EEEEEEEEEEE--H--HHHHHHHHHHHHHHH-----EEEEEE--------EEEEEE
```

| V | Rank | Seed | Score | Length | Start | End | Pfam | Title | Compound |
|---|---|---|---|---|---|---|---|---|---|

Show top 100 results

***Figure 8.2. The*** Fold Recognition ***panel after*** Run SSP***.***

### 8.3.3 Editing or Deleting SSPs

You can use your expert knowledge to improve the query's SSP profile by deleting or editing SSPs you believe are incorrect.

To delete an SSP, right-click on the SSP in the sequence viewer and choose Delete from the shortcut menu.

**To edit an SSP:**

1. Click the Edit SSP toolbar button:
2. Highlight the part of the SSP you want to change.
3. Type the character `e`, `h`, or `-`.

   E represents strand, H represents helix, and – represents loop.

The highlighted region of the prediction is changed to your choice.

### 8.3.4 Resetting SSPs

To retrieve an unmodified SSP after editing, click Run SSP. This immediately resets the SSP (but does not rerun the prediction programs).

## 8.4 Searching for Templates

When you are satisfied with the SSP profile for the query, click Search to run the Fold Recognition program. You may prefer to change the search options described in the following section before doing so.

### 8.4.1 Search Program Options

#### Display of Results

Show Top *N* Results

By default, the 100 top-ranked templates found are shown in the Templates table. This number can be changed in the Show Top *N* Results text box before clicking Search.

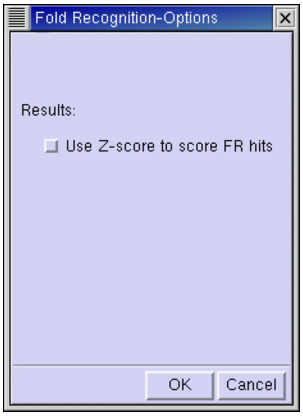


***Figure 8.3. The*** Fold Recognition–Options ***dialog box.***

## Z-Scoring

Searching can be done with or without Z-scoring. The Z-score shows how significant a match is relative to a random match. It is calculated by shuffling the query sequence. See Section 10.2 on page 98 for more information on Z-score filtering.

You may want to turn on Z-scoring if the query protein is any of the following:

- Less than 120 residues long
- More than 90 percent alpha (helical)
- More than 90 percent beta (strand)

Click Options to open the Fold Recognition–Options dialog box and turn on Z-scoring.

**Note:** Running the search program with Z-scoring on a query with 300 residues typically requires three to four hours. Without Z-scoring, running a search on a protein of similar length typically takes only 10-15 minutes.

## 8.4.2 Search Program Technical Details

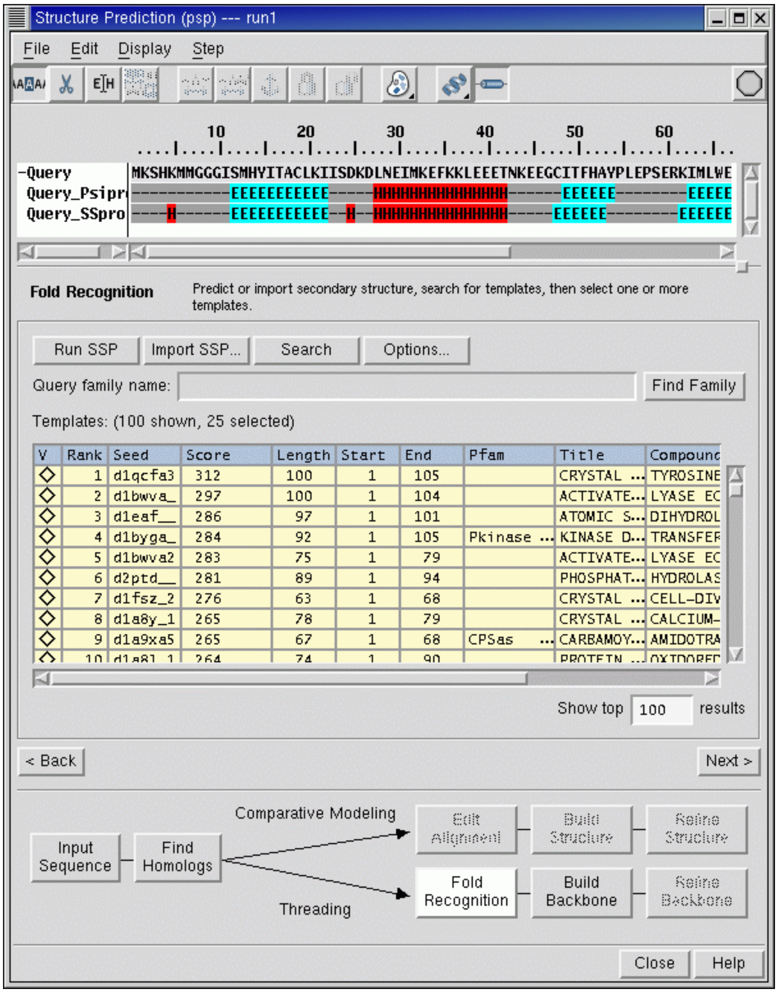
The search program finds potential templates for model building by searching a database of structure folds (SCOP domains) generated from the PDB using secondary structure information.

Profile-sequence matching and composite secondary structure information are the same as in the Align program used in Edit Alignment in the Comparative Modeling path. The search program used in the Threading path has two additional features:

- Weighting is adjusted by degree of structural conservation inside a family of templates.

Each residue in the template sequence is defined as conserved or variable according to Multiple Structure Alignment (MSTRA) of the template with its structural neighbors. When aligning conserved residues, higher weight is given for matching and higher penalty is given for opening gaps, and vice versa for variable residues.

- Optional Z-scoring, as described above and discussed in more detail in Section 10.2 on page 98.



```
                  10        20        30        40        50        60
              ....|....|....|....|....|....|....|....|....|....|....|....|..
-Query        MKSHKMMGGGISMHYITACLKIISDKDLNEIMKEFKKLEEETNKEEGCITFHAYPLEPSERKIMLWE
 Query_Psipr  ----------EEEEEEEEEEE-----HHHHHHHHHHHHHHH------EEEEEE--------EEEEE
 Query_SSpro  ----H------EEEEEEEEEEE--H--HHHHHHHHHHHHHHH-----EEEEEE--------EEEEEE
```

Predict or import secondary structure, search for templates, then select one or more templates.

Query family name:

Templates: (100 shown, 25 selected)

| V | Rank | Seed | Score | Length | Start | End | Pfam | Title | Compounc |
|---|---|---|---|---|---|---|---|---|---|
| ◇ | 1 | d1qcfa3 | 312 | 100 | 1 | 105 | | CRYSTAL ... | TYROSINE |
| ◇ | 2 | d1bwva_ | 297 | 100 | 1 | 104 | | ACTIVATE... | LYASE EC |
| ◇ | 3 | d1eaf__ | 286 | 97 | 1 | 101 | | ATOMIC S... | DIHYDROL |
| ◇ | 4 | d1byga_ | 284 | 92 | 1 | 105 | Pkinase ... | KINASE D... | TRANSFER |
| ◇ | 5 | d1bwva2 | 283 | 75 | 1 | 79 | | ACTIVATE... | LYASE EC |
| ◇ | 6 | d2ptd__ | 281 | 89 | 1 | 94 | | PHOSPHAT... | HYDROLAS |
| ◇ | 7 | d1fsz_2 | 276 | 63 | 1 | 68 | | CRYSTAL ... | CELL-DIV |
| ◇ | 8 | d1a8y_1 | 265 | 78 | 1 | 79 | | CRYSTAL ... | CALCIUM- |
| ◇ | 9 | d1a9xa5 | 265 | 67 | 1 | 68 | CPSas ... | CARBAMOY... | AMIDOTRA |
| ◇ | 10 | d1a81_1 | 264 | 74 | 1 | 90 | | PROTEIN ... | OXIDORED |

Show top 100 results

*Figure 8.4.* ***The*** Fold Recognition ***panel with*** Search ***and*** Find Family ***results.***

### 8.4.3 Search Output

When the search is complete, the Templates table displays candidate templates, ranked from the closest homolog to the least close. This preliminary ranking is based on a scoring function that combines sequence, secondary structure, and degree of structural conservation. If Z-scoring is on when Search is launched, rankings take Z-score into account as well.

While it is possible that the top-ranked template will yield good results, it is recommended that the top 25 to 40 templates be selected to pass on to the Build Backbone step. By default, the top 25 templates are selected.

To select more, fewer, or different templates, hold down the CTRL key or the SHIFT key and select the appropriate rows in the table. To sort the templates by properties other than rank, click the appropriate column heading.

Optionally, if you have not already searched for the family of the query, you can click Find Family to run HMMER/Pfam. The query family name found, with an E-value in parentheses, appears in the Query family name text box. The sequence viewer shows the match between the query and the consensus sequence of the family, as described in Section 4.2.2 on page 47. If a potentially meaningful (E-value under 1) family identification is found, it may help you select different or additional seed templates to bring forward to Build Backbone.

To view a template colored by secondary structure, click the check box in the V (Visualization) column. The template appears in the sequence viewer, and its secondary structure can be compared to the SSP profile of the query.

**Note:** The query-template "alignments" that can be visualized in this step are provided only as a guide in identifying seed templates. Gaps or other unfavorable features will not affect the quality of the models built in subsequent steps.

When you are satisfied with the selected templates, click Next to continue to the Build Backbone step.

# Threading: Build Backbone

## 9.1 Composite Templates

The Build Backbone step is designed to generate correctly folded backbone structures of the query, based on templates of very low sequence identity (typically <15%), when no closer homologs are available.

If no template is more than remotely related to the query, even if structural similarity is all that is being considered, it is difficult to find a single template similar to the entire query. The creation of aligned composite templates is designed to overcome this problem. Composite templates are generated by mixing and matching the variable regions in the seed family with sequences from other members of the family that may be a better match for the query.

Each template selected in the previous step is used as a seed to generate thousands of composite templates, each with dissimilar regions of the seed template replaced with sub-sequences from its structural neighbors. The query is then aligned to these composite templates, scored, and then clustered to reduce redundancy and filtered based on the score. High-ranking composite templates can then be selected for the next step, Refine Backbone.

## 9.2 Job Control and Output Options

The Structures table in the Build Backbone step is initially empty (see Figure 9.1). The seed template selections made in the previous step, Fold Recognition, are used by the Build Backbone program to generate structures and populate the table. For each seed template, Build Backbone runs a job that returns a number of candidate structures. The default maximum structures per seed template is 20. These jobs can be run serially, if only one processor is available, or they can be run in parallel on multiprocessor machines or clusters. When you click the Build button, a dialog box prompts you to specify the number of jobs to run in parallel. You can change the host machine and login for the Build Backbone step by clicking the Edit button to open the Job Options dialog box.

By default, each job returns up to 20 composite templates. However, the number and variety of composite templates returned can be controlled using the Options dialog box. Click the Options button and choose among the following options:

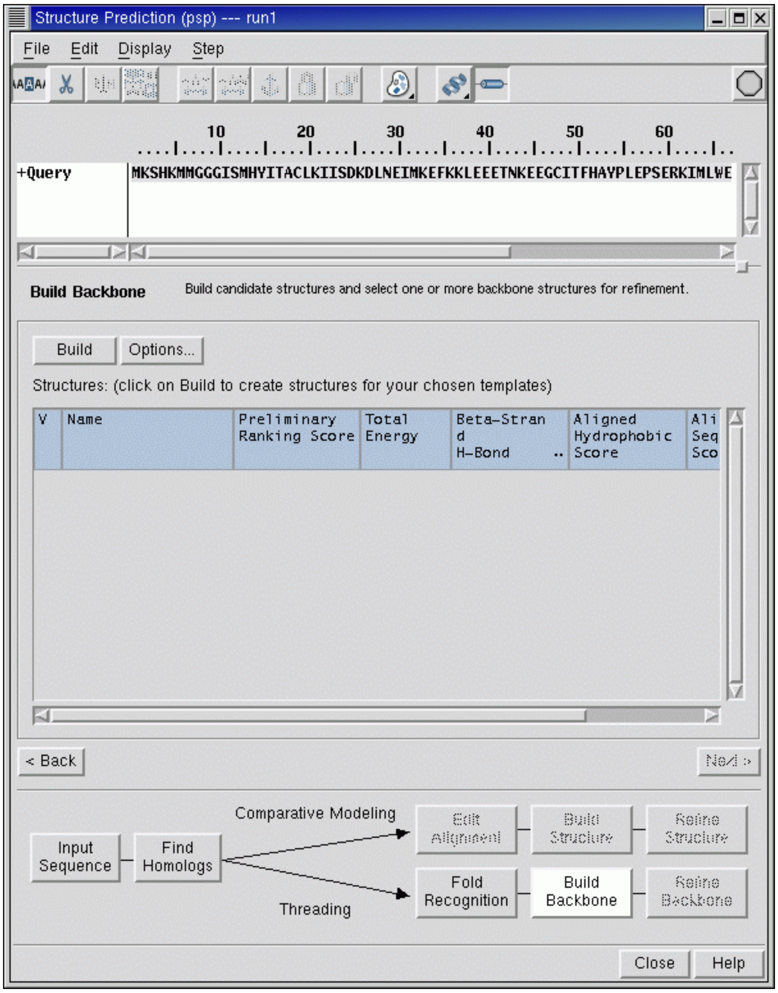
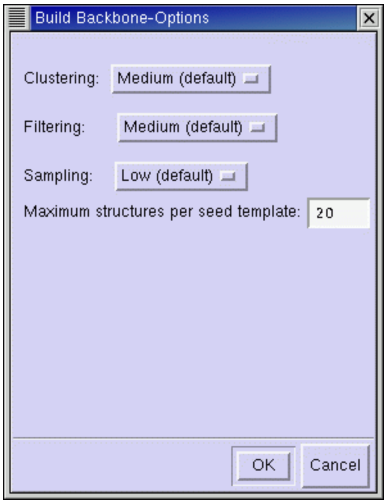

Structure Prediction (psp) --- run1
File Edit Display Step
+Query
MKSHKMMGGGISMHYITACLKIISDKDLNEIMKEFKKLEEETNKEEGCITFHAYPLEPSERKIMLWE
Build Backbone
Build candidate structures and select one or more backbone structures for refinement.
Build
Options...
Structures: (click on Build to create structures for your chosen templates)
V
Name
Preliminary Ranking Score
Total Energy
Beta-Strand H-Bond
Aligned Hydrophobic Score
Ali Seq Sco
< Back
Next >
Comparative Modeling
Threading
Input Sequence
Find Homologs
Edit Alignment
Build Structure
Refine Structure
Fold Recognition
Build Backbone
Refine Backbone
Close
Help

***Figure 9.1. The*** Build Backbone ***step, initial view.***

Clustering

Determines how different two structures must be in order to be treated independently by the filtering module. Higher levels are faster but less accurate, and return fewer structures. The default option is Medium. The options are:

- None: All structures are treated independently, resulting in a large number of composite templates returned.
- Low: Some nearly-redundant structures are not treated independently.



Clustering: Medium (default)

Filtering: Medium (default)

Sampling: Low (default)

Maximum structures per seed template: 20

***Figure 9.2. The*** Build Backbone-Options ***dialog box.***

- Medium (default).
- High: Fewer structures are treated independently.

Filtering

- Low: More composite templates are kept—slower but more accurate calculation.
- Medium (default).
- High: Fewer composite templates are kept—faster but less accurate calculation.

Sampling

- Low (default): Generates up to 5,000 composite templates.
- Medium: Generates up to 25,000 composite templates.
- High: Generates up to 50,000 composite templates.

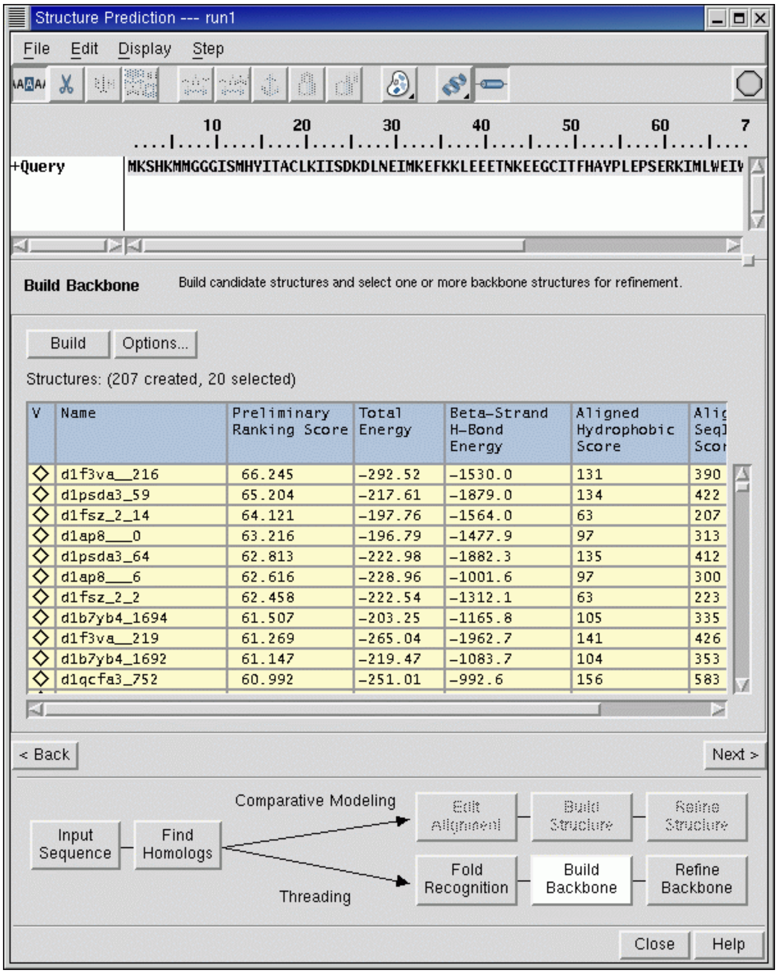
Higher sampling levels are slower but more accurate.

Maximum structures per seed template

The default is 20.

## 9.3 Output Structures

When the Build jobs are finished, the results appear in the Structures table. Each structure in the table corresponds to a preliminary query/composite template alignment. The structure Name indicates the origin of the composite template in a seed template and its structural family.



Structures: (207 created, 20 selected)

| V | Name | Preliminary Ranking Score | Total Energy | Beta-Strand H-Bond Energy | Aligned Hydrophobic Score | Alig SeqI Scor |
|---|---|---|---|---|---|---|
| ◇ | d1f3va__216 | 66.245 | -292.52 | -1530.0 | 131 | 390 |
| ◇ | d1psda3_59 | 65.204 | -217.61 | -1879.0 | 134 | 422 |
| ◇ | d1fsz_2_14 | 64.121 | -197.76 | -1564.0 | 63 | 207 |
| ◇ | d1ap8___0 | 63.216 | -196.79 | -1477.9 | 97 | 313 |
| ◇ | d1psda3_64 | 62.813 | -222.98 | -1882.3 | 135 | 412 |
| ◇ | d1ap8___6 | 62.616 | -228.96 | -1001.6 | 97 | 300 |
| ◇ | d1fsz_2_2 | 62.458 | -222.54 | -1312.1 | 63 | 223 |
| ◇ | d1b7yb4_1694 | 61.507 | -203.25 | -1165.8 | 105 | 335 |
| ◇ | d1f3va__219 | 61.269 | -265.04 | -1962.7 | 141 | 426 |
| ◇ | d1b7yb4_1692 | 61.147 | -219.47 | -1083.7 | 104 | 353 |
| ◇ | d1qcfa3_752 | 60.992 | -251.01 | -992.6 | 156 | 583 |

***Figure 9.3. The*** Build Backbone ***step with building results.***

To help you choose which candidate structures (query/composite template alignments) to bring into Refine Backbone, each is given a Preliminary Ranking Score. By default, the 20 highest-scoring structures are selected for refinement.

*Table 9.1. Description of the Structures table columns.*

| Column | Description |
|---|---|
| V | Visualization controls. By default, none of the structures is shown in the Workspace. Backbone models that have not yet been refined are preliminary and may display nonphysical structures. To visualize a structure in the Workspace, select the checkbox in this column. |
| Name | Indicates the origin of the composite template in a seed template and its structural family, with a number to distinguish different templates derived from the same seed template. |
| Preliminary Ranking Score | This is a composite score which produces the best ranking to use for selecting backbone structures for refinement. You can sort on any of the component scores by clicking the column header. |
| Total Energy | A weighted sum of the scores below. |
| Beta-Strand H-Bond Energy | H-bond distance constraint energy for the beta-strands only. |
| Aligned Hydrophobic Score | Alignment score from hydrophobicity matching. |
| Aligned SeqID Score | Alignment score from profile-sequence matching. |
| Aligned SS Score | Alignment score from secondary structure matching. |

You can inspect a candidate structure/preliminary alignment in the Maestro Workspace, colored by secondary structure. Click the check box in the V (Visualization) column for the structure you want to view (only one can be displayed at a time).

This will also display the composite template sequence alignment being visualized in the sequence viewer. If only one sequence is shown, then that composite template is derived solely from the seed template itself and will have the name of the seed template PDB file in the leftmost column.

If more than one sequence is shown, then the seed template is listed first, followed by the other templates in its family that contribute to the variable regions comprising the composite template. The PDB ID of each contributing structure appears in the leftmost column. The residues contributing to the composite template are highlighted. The final sequence is the resulting composite template sequence.

**Note:** These alignments or preliminary backbone models are highly incomplete, and are available to be viewed strictly as a guide in selecting candidates.

You may select as many structures as you like by holding down the CTRL key or holding down the SHIFT key and clicking on rows. The default (the top 20) usually includes the best structures for refinement. When your selections are complete, click Next to proceed to the Refine Backbone step.

## 9.4 Build Backbone: Technical Details

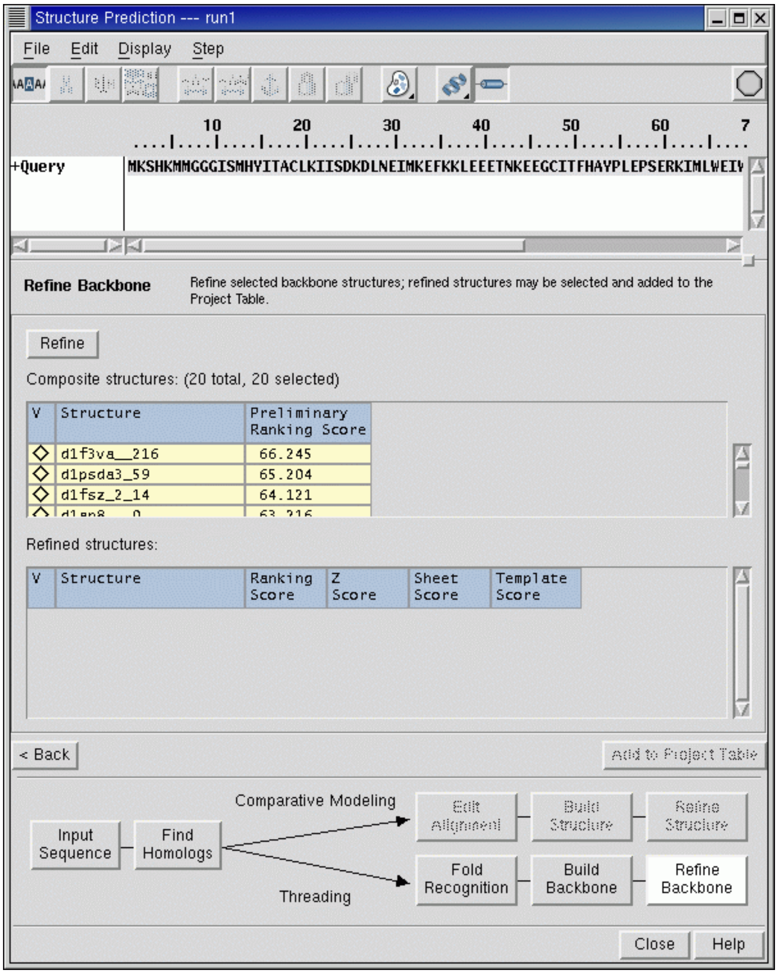
The Build Backbone program generates a set of composite templates by replacing those regions of the original ("seed") template that are dissimilar to the query with sub-sequences from structural neighbors in its family that might be a better match. This process is done systematically as follows:

- Conserved and variable regions of a template are identified based on multiple structure alignment (MSTRA), and conserved regions of the seed template are used as anchors to build the composite templates.
- Sub-sequences from different proteins in a variable region are enumerated to find replacements for (or insertions into) corresponding parts of the seed template.
- Each composite template is aligned to the query in the same way as an ordinary template. The features of Single Template Alignment (STA) described in Edit Alignment and Fold Recognition also apply here: profile sequence matching, composite secondary structure prediction for the query, and weighting adjusted by degree of conservation.
- In addition, the predicted secondary structure of the query can be overwritten by the template (with a penalty) if it aligned to a highly conserved region in the template.
- Because the composite template contains pieces of proteins from different coordinate frames, the initial model from the composite template alignment is built by constructing a rotation matrix that rotates the coordinate frames of the templates in the Multiple Structure Alignment (MSTRA) into a common frame.
- Structures built from composite template alignment are evaluated by tertiary folding potentials in addition to alignment scores. A filter module is used to reduce the number of structures that must be considered for future refinement. After filtering and initial refinement, a scoring function that incorporates alignment scores (sequence, secondary structure) and H-bonding energy is used to further reduce the number of structures going into extensive refinement.

# Threading: Refine Backbone

## 10.1 Overview of the Refine Backbone Step

The Refine Backbone step begins with the composite templates selected in Build Backbone listed in the Composite structures table. By default, these are the 20 top-ranked structures in



Refine selected backbone structures; refined structures may be selected and added to the Project Table.

Composite structures: (20 total, 20 selected)

| V | Structure | Preliminary Ranking Score |
|---|---|---|
| ◇ | d1f3va__216 | 66.245 |
| ◇ | d1psda3_59 | 65.204 |
| ◇ | d1fsz_2_14 | 64.121 |

Refined structures:

| V | Structure | Ranking Score | Z Score | Sheet Score | Template Score |
|---|---|---|---|---|---|

*Figure 10.1. The* Refine Backbone *panel, initial view.*

Build Backbone (see Figure 10.1). Each of these composite structures was generated from a Fold Recognition template and an alignment in the Build Backbone step. The composite structures are incomplete, containing only aligned residues without side chains, and are given only a preliminary ranking score. The secondary structure is assigned, and does not necessarily match the actual coordinates. Refining the composite structures produces complete backbone structures and more accurate ranking scores. One or more refined backbone structures can be chosen for addition to the Project Table. This completes the Threading Path run.

**To perform the Refine Backbone step:**

- Select the composite structures you want to refine, then click Refine.

**Note:** The structures selected in Build Backbone (by default, 20) are still selected when you enter Refine Backbone, and all twenty refinement jobs are run when you click Refine. To run the refinement jobs in smaller groups, change your selections appropriately.

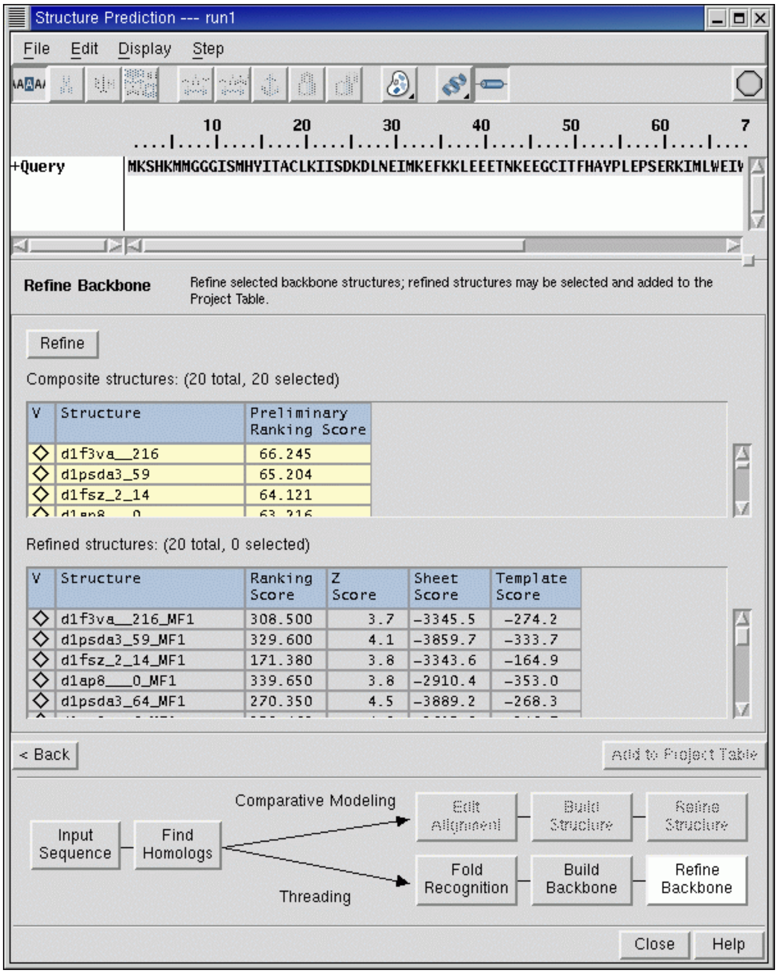
When refinement of a structure is complete, it is added to the Refined structures table. Structures can be visualized and compared in the Workspace by checking the V (Visualization) column.

Sort the table by Ranking Score. In most cases the structure with the highest score is the best structure.

## 10.2 Scores of Refined Structures

Each output structure is given four scores, as shown in the Refined structures table in Figure 10.2:

1. Ranking Score: This is the result of the Global Scoring Function, incorporating the three other scores listed (Z-Score, Sheet Score, and Template Score), and is the number that best represents the overall quality of the structure. Ranking Score is designed to be transferable between templates, allowing you to select the best predictions among all simulations run for a given query. This is required because the actual simulation energies are dependent on secondary structure assignments, domain boundaries, and so on, and cannot reliably be compared between different alignments. See Section 10.3 on page 100 for details of the function that generates the Ranking Score. The Ranking Score is also listed as the Prime Threading Score in the Maestro Project Table.
2. Z Score: Sequence-shuffled Z-Score. This is the statistical potential energy Z-Score of the query sequence compared to randomly shuffled sequences of the same composition evaluated for the same structure. It is a measure of the sequence-structure compatibility.



MKSHKMMGGGISMHYITACLKIISDKDLNEIMKEFKKLEEETNKEEGCITFHAYPLEPSERKIMLWEIV

**Refine Backbone** Refine selected backbone structures; refined structures may be selected and added to the Project Table.

Refine

Composite structures: (20 total, 20 selected)

| V | Structure | Preliminary Ranking Score |
|---|---|---|
| ◇ | d1f3va__216 | 66.245 |
| ◇ | d1psda3_59 | 65.204 |
| ◇ | d1fsz_2_14 | 64.121 |

Refined structures: (20 total, 0 selected)

| V | Structure | Ranking Score | Z Score | Sheet Score | Template Score |
|---|---|---|---|---|---|
| ◇ | d1f3va__216_MF1 | 308.500 | 3.7 | -3345.5 | -274.2 |
| ◇ | d1psda3_59_MF1 | 329.600 | 4.1 | -3859.7 | -333.7 |
| ◇ | d1fsz_2_14_MF1 | 171.380 | 3.8 | -3343.6 | -164.9 |
| ◇ | d1ap8___0_MF1 | 339.650 | 3.8 | -2910.4 | -353.0 |
| ◇ | d1psda3_64_MF1 | 270.350 | 4.5 | -3889.2 | -268.3 |

***Figure 10.2. The*** Refine Backbone ***step, after refinement.***

3. Sheet Score: This is the score associated with the H-bond and local template constraints derived from the strand-pairing. For the regular minimization protocols it is normalized in such a way that the maximum value is 2.

4. Template Score: This is the score associated with the superposition of the final structure on the original composite template. This reflects the quality of the alignment, but also the relaxation of the structure away from the template during energy minimization and therefore also the compatibility of the template coordinates and the other constraints.

## 10.3 The Ranking Score and the Global Scoring Function

The Global Scoring Function program that produces the Ranking Score is run as a post-processing step. It takes several "descriptors" (values of certain calculated properties) and applies various non-linear transformations and normalizations before producing a linear combination of the transformed values.

Its input comes from several sources:

1. Carried over from previous steps:
   - Aligned SeqID Score (from Build Backbone)
   - Aligned SS Score (from Build Backbone)
   - FR Score (from Fold Recognition)
2. Generated by the Refine Backbone module:
   - Total constraint energy: all amino-acid-independent terms in the simulation energy
   - Hydrogen bond energy

   and as mentioned in Section 10.2:

   - Sheet Score
   - Template Score
   - Z-Score
3. Calculated separately in post-processing:
   - Buried charged residues
   - Hydrophobic patches in sheets
   - Unpaired strand count

## 10.4 Automatically Generated Constraints

Pre-processing is carried out automatically to generate constraints to be used by the simulation. The constraints are as follows:

- The input structure is used as a constraint, with the RMS superposition of the simulated structure C$\alpha$ atoms on the template C$\alpha$ atoms used as a term in the scoring function.
- Angle constraints are applied based on the assigned secondary structure. Helices are held rigid to within 10°, strands have a restraint force on $\phi$ and $\psi$, and loops can move freely.
- The input structure is scanned for all possible hydrogen bonds using geometrical/energetic criteria for residues which are present. The list of H-bonds is parsed for parallel /

antiparallel beta sheet patterns and the longest "non-conflicting" ladders are used to generate a strand pairing file.

- The strand pairing file is used to automatically generate H-bond constraints in the simulation and also to generate local (two strands at a time) coordinate templates to establish a "sheet-like" overall structure.
- Domain boundaries are read from the alignment and/or constructed from multiple files. These are used to truncate inter-domain potential interactions in the simulation.

## 10.5 Adding Side Chains

Side chains can be added to a backbone structure. To accomplish this, select a structure from the table and click Add to Project Table. Include this structure in the Workspace, then choose Refinement from the Prime menu, opening the Prime–Refinement stand-alone module. Click Load Defaults, select Task: Predict side chains, click the All button, and then click Run.

## 10.6 Adding to the Project Table

When the modeling and refinement are complete, you can save the final structure to a Maestro project by clicking Add to Project Table. The Project Table is displayed with the new backbone structure appended. Now the model structure can be used in other modules or exported to other file formats.

**To export a Project Table entry to a file of another format:**

1. In the Project Table, select the structure you want to export, then choose Export from the Table menu.
2. In the Export panel:
   a. Select the format (Maestro, MMOD, PDB, Mol2, SD).
   b. Enter a file name.
   c. Make sure the Structure Source To Be Exported option is set to Selected Entries.
   d. Click the Export button.

# Prime–Refinement

Prime–Refinement is a stand-alone module used to refine protein structures from the Maestro Workspace. The methods used in the Refine Structure step of Prime Comparative Modeling can thereby be applied to protein structures from any source, independent of the Prime–Structure Prediction workflow. It runs independently of Prime–SP from its own Maestro panel, the Refinement panel. The Refinement panel offers the same refinement protocols (loop refinement, side-chain prediction, and minimization) that are available in the Refine Structure step in the Prime–SP Comparative Modeling Path. It also offers a fourth task, a single-point energy calculation at the current geometry of the model structure. In the Comparative Modeling Path of Prime–SP, this energy is calculated automatically when Build Structure finishes.

In many cases, Prime–Refinement is used on protein structures from sources other than Prime, but it can also be used with structures added to a project from the Refine Structure step in the Prime–SP module. For example, if you have added a water molecule to the binding site of the protein produced by Refine Structure, you can use Prime–Refinement on the composite entry. Likewise, you can refine structres from the Refine Backbone step. Structures from Refine Backbone do not have side chains, but if you choose side-chain prediction for such a structure, side chains are automatically added and then predicted.

Prime–Refinement can also refine structures with covalently bound ligands. The ligands can be multiply connected, covalently bound to each other and to the protein. This capability allows the refinement of proteins with phosphorylated residues or attached sugars, for example.

## 11.1 Preparing Structures for Refinement

Prime–Refinement has a great deal of flexibility in the types of structures it can handle and can fix many structural problems. For standard residues, Prime–Refinement can fix formal charges and bond orders, and correct disparities between the sequence and the structure. For example, if a residue has the coordinates of ALA but is called SER, the 3HB will be ignored and the OG and HG added during refinement. The standard residues are the 20 canonical amino acids; ACE and NMA; HOH; CYX (disulfide); and the acid/base variants ASH/AS1 (ASP), GLH/GL1 (GLU), ARN (ARG), LYN (LYS), HIE/HIP/HID (HIS), CYT (CYS), SRO (SER), TYO (TYR).

Prime–Refinement has certain conditions that must be met by the input structures for a job to be run successfully. Some of these conditions only apply to covalently bound ligands.

For all kinds of structures, the following condition must be met:

- The structure must be an all-atom structure.

  This condition applies to the protein and any ligand, waters or cofactors present in the structure: all hydrogens must be present.

  You can add hydrogens to a structure by displaying the structure in the Workspace and double-clicking the Add hydrogens button in the toolbar.

  However, for structures containing nonstandard residues, you must correct bond orders and formal charges first.

For structures with nonstandard residues, the following conditions must be met:

- No residue can consist of disconnected pieces without formal bonds.

  This means, among other things, that metals should not have formal bonds to the rest of the structure, and should be treated as ionic. For example, in hemes the Fe must be a separate residue with a +2 formal charge and no bonds, and the “bonded” N atoms must have a –1 formal charge.

- Bound residues must contain at least 3 atoms between bonds to other residues.

  This means that, for example, a dihedral angle cannot span more than 2 residues.

- Chemically distinct ligands must have different residue names.

  Multiple occurences of the same ligand (such as NAG bound to ASN) are considered chemically identical, but other occurences of NAG with different bonds must be renamed (such as NAGs bound to each other in a chain). This also applies to DNA treated as a ligand: all nucleotide bases in the interior of the chain are equivalent, but the 3' and 5' bases must be relabeled.

  The residue names XXX, YYY, and ZZZ are reserved for the protein backbone and cannot be used for ligands.

- Nonstandard residues cannot appear at the chain termini.

  The only recognized termini are ACE, NMA, and the 20 standard amino acids in charged form; but disulfide bonds to the terminal residues are permitted.

- Bond orders and formal charges must be corrected.

  For nonstandard residues the supplied structure will be used, and you must check for correct bond orders and formal charges.

When you rename residues, you should be aware that OPLS_2000 parameters are used for the standard residues, whereas nonstandard residues are treated with OPLS_2001 parameters. This means that slight differences in structure can occur, depending on how a residue is named: for example, changing the name of a canonical structure; or treating a large ligand as separate parts or as a single residue; or treating a modified side chain as a single nonstandard residue or as a standard residue stem with a covalently-bound ligand.

Proteins can also be used as ligands, and the same conditions apply. No special treatment is needed for two distinct chains connected by a disulfide or similar sidechain linkage. If the "ligand" chain has nonstandard terminal groups or is connected to the main chain by its backbone, two steps are required to make sure that the "ligand" chain is treated correctly:

1. Rename standard residues appearing in the ligand to nonstandard names.
2. Rename the atoms in inter-residue C–N bonds between renamed (ligand) residues. For example, change "_C__ "to "_C*_", where the underscores represent spaces. Other atoms can keep their PDB names; only one of C or N needs to be changed.

If your structure does not meet one or more of the conditions outlined above, you must fix it before you can successfully run a refinement. Some of the problems only become apparent when you have run a refinement job, so it can be a useful diagnostic procedcure to run a Prime energy calculation first. This job is quick, and produces warning in the log file, which you can check in the Monitor panel.

Procedures for fixing bond orders, formal charges, and atom names are given below. To use these procedures, you must have the Build panel open, which you can do by clicking Open/close Build panel the on the main toolbar.

**To assign bond orders:**

1. Choose Assign Bond Orders from the Tools menu.
2. Inspect the residues in the structure for any remaining bond orders that are incorrect.
3. If there are still incorrect bond orders, click the Increment bond order or Decrement bond order button on the Build panel toolbar.
4. Click the bonds in the Workspace structure that needs correction.

**To view and change formal charges:**

1. From the Label atoms button menu on the main toolbar, choose Formal Charge.

   Charges are displayed for atoms that are charged.

2. Click the Increment formal charge or Decrement formal charge button on the Build panel toolbar.

3. Click on an incorrectly charged atom in the Workspace until its charge is correct.

   The formal charge is incremented or decremented by one unit for each click.

**To change PDB atom names:**

1. From the Label atoms button menu on the main toolbar, choose PDB Atom Name.

2. In the Atom Properties folder, choose PDB Atom Name from the Property option menu.
3. Ensure that Pick is selected in the Apply PDB atom name section, and that Atoms is chosen from the Pick option menu.
4. Zoom in on one of the residues (middle+right mouse buttons).
5. For each atom that needs renaming, enter the desired PDB atom name in the PDB atom name text box, then click on the atom.
6. Repeat Step 4 and Step 5 for each residue whose atoms need renaming.

**To change residue names:**

1. From the Label atoms button menu on the main toolbar, choose Residue information.

2. In the Residue Properties folder, choose Residue Name from the Property option menu.
3. Ensure that Pick is selected in the Apply residue PDB name section, and that Residues is chosen from the Pick option menu.
4. For each residue that needs renaming, enter the desired PDB name in the Residue PDB name text box, then click on the residue.

   You might need to zoom in to select the residues (middle+right mouse buttons).

## 11.2 Using the Refinement Panel

The four tasks that can be performed in the Refinement panel are single-point energy calculation, loop refinement, side-chain prediction, and minimization. These tasks are discussed in the following sections. The general procedure for running a structure refinement is given below.

**To run structure refinement tasks:**

1. Display the structure you want to refine in the Workspace.
2. Choose Refinement from the Prime submenu of the Maestro Applications menu.

   The Refinement panel is displayed.
3. Select a task from the Task menu.

   The default task is Refine loops.
4. Select the region for refinement, as appropriate.

   When you have selected a region for refinement, the Start and Write buttons become available.
5. Click Start.

   The Start dialog box is displayed. If you want to run the job later, using `refinestruct`, you can click Write to create the input files.
6. Make changes to the job settings, as appropriate.

   The settings for running refinement jobs are shown in the upper part of the Refinement panel: Job (base file name), Login name, and from the Hosts options, the name of the current host machine.
7. Click Start.

   The refinement job begins and the Job Monitor panel is displayed.

The Start and Write buttons remain dimmed until you have specified some part of the structure for refinement. When you have made a selection, the Start and Write buttons become available. The Write button writes the input file for the job, which you can run at a later time

When selecting regions for refinement, you may want to look at the `.psane` file generated for the structure being refined. Information about possible structural problems, as in the Protein Reporting Panel, can be found here. A new `.psane` file is generated after each successful refinement task.

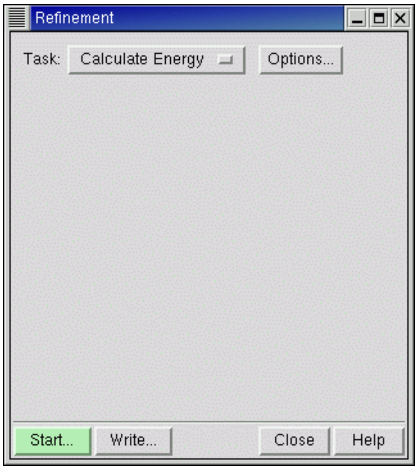


***Figure 11.1. The*** Calculate Energy ***task in the*** Refinement ***panel.***

## 11.3 Calculating the Prime Energy

To perform a single-point molecular mechanics energy calculation, choose Calculate Energy from the Task option menu. The calculation uses the OPLS_2000 all-atom force field (protein-optimized) for protein residues and OPLS_2001 for ligands and cofactors.

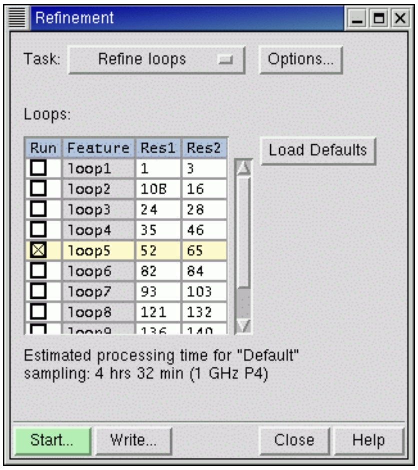
## 11.4 Refining Loops

Prime–Refinement is capable of refining loop structures of various lengths, and provides algorithms for different loop lengths. In addition, loops whose structure affects other loops can be cooperatively refined in pairs.

**To refine one or more loops serially:**

1. Choose Refine Loops from the Task menu.
2. Click Load Defaults.

   The loop features of the Workspace structure appear in the Loops table. For each loop, the table includes a Run check box, a feature name such as loop1, and the numbers of the first (Res1) and last (Res2) residues in the loop. When you click a row in the table to select a loop, the Workspace shows the loop residues highlighted in yellow. Note that loops of one or two residues are not listed, as they are unsuitable for the refinement protocol.
3. Select a loop by clicking its name in the Feature column.



Task: Refine loops  Options...

Loops:

| Run | Feature | Res1 | Res2 |
|---|---|---|---|
| ☐ | loop1 | 1 | 3 |
| ☐ | loop2 | 10B | 16 |
| ☐ | loop3 | 24 | 28 |
| ☐ | loop4 | 35 | 46 |
| ☒ | loop5 | 52 | 65 |
| ☐ | loop6 | 82 | 84 |
| ☐ | loop7 | 93 | 103 |
| ☐ | loop8 | 121 | 132 |
| ☐ | loop9 | 136 | 140 |

Load Defaults

Estimated processing time for "Default" sampling: 4 hrs 32 min (1 GHz P4)

***Figure 11.2. The*** Refine loops ***task in the*** Refinement ***panel.***

The loop row is highlighted in yellow. The default settings for the loop are displayed.

4. If necessary, shorten the loop region to be refined by changing the first and last residue numbers (Res1 and Res2).

   The time required to refine a loop scales approximately linearly with the length of the loop, and the sampling options recommended for loops longer than five residues are also more time-intensive.

5. (Optional) Click Options to select a sampling method and make settings for this loop in the Refine Structure-Options dialog box.

   If the loop is longer than five residues, you should select a sampling method other than the default.

   See Section 11.7.3 for information on the loop refinement options.

6. To designate this loop for refinement, select the check box in the Run column.

   The Start and Write buttons becomes available.

7. Repeat Step 3 through Step 6 for each loop you want to refine.

8. Click Start.

   The Start dialog box opens.

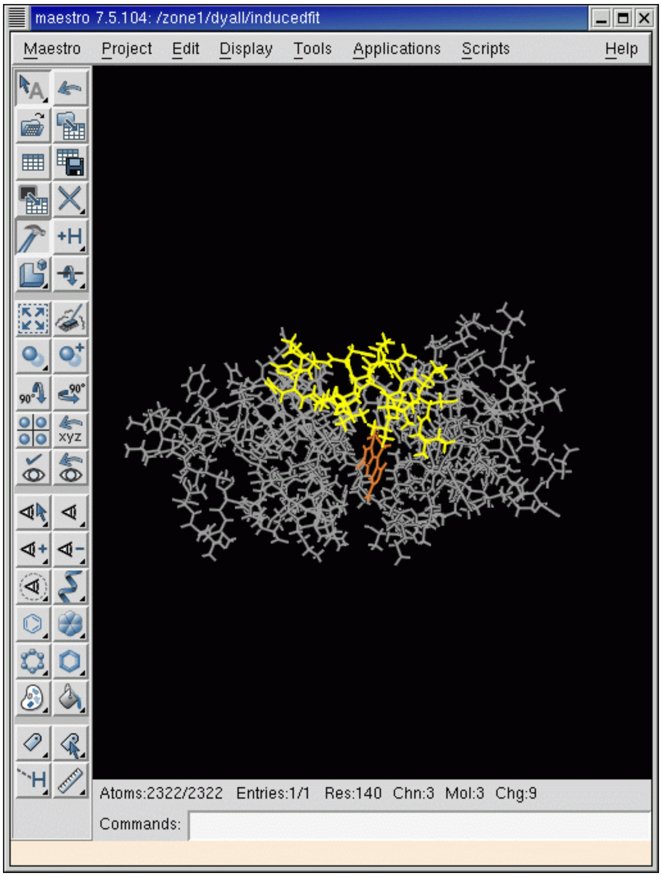
9. Make any job settings, then click Start.

   The loop refinement job is started.

When you choose several loops for refinement, they are refined in series. The first selected loop is refined, the best scoring structure for that loop is determined, that structure is used in refining the next loop, and so on. When all loops have been refined, one final structure is returned. If only one loop is refined, the four highest scoring structures are returned by default.

If you want to refine two loops cooperatively, you should replace Step 3 through Step 7 above with the following steps:

1. Click the Run column for the two loops you want to refine cooperatively.
2. If necessary, shorten the loop region to be refined by changing the first and last residue numbers (Res1 and Res2).
3. Click Options to open the Refine Structure-Options dialog box.


maestro 7.5.104: /zone1/dyall/inducedfit
Maestro Project Edit Display Tools Applications Scripts Help
Commands:

Atoms:2322/2322 Entries:1/1 Res:140 Chn:3 Mol:3 Chg:9

**Figure 11.3. The Workspace with the residues selected for loop refinement highlighted.**

4. Select Cooperative loop sampling in the Loop refinement section.
5. Set any other options, then click OK.

For technical details about loop refinement, see Section 7.8.1 on page 80.

When a loop refinement begins, a validation program checks the loop features of the structure. Apparent errors are reported in a warning dialog box. You may choose Run All Features, Run Only Valid Features, or Cancel. It is recommended that you make a note of the invalid features, click Cancel, and correct the structure if appropriate.

## 11.5 Predicting Side Chains

**To predict side-chain conformations:**

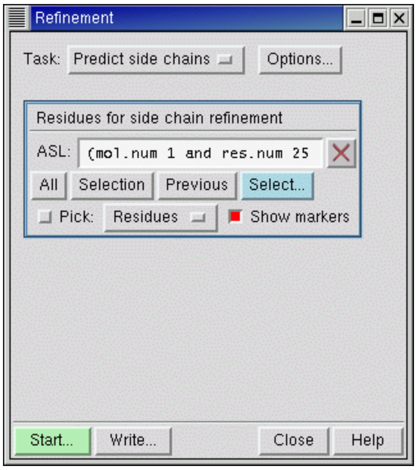
1. Select Predict side chains from the Task options menu.

   The panel displays atom selection options under the heading Residues for side chain refinement.

2. Use the atom selection options to specify the residues for which you want to predict or refine side-chain conformations, as described in Section 2.6 on page 23.

   For more information, see the online help or Chapter 5 of the *Maestro User Manual*.

3. Click Options to change the Number of structures to return from the default, which is to return a single side-chain prediction.
4. Click the Run button to start the side-chain prediction job.


Task: Predict side chains

Residues for side chain refinement

ASL: (mol.num 1 and res.num 25

**Figure 11.4.** ***The*** Predict side chains ***task in the*** Refinement ***panel.***

You can also use this process to add side chains to a backbone structure from Refine Backbone.

For technical details about side-chain prediction, see Section 7.8.3 on page 81.

## 11.6 Minimizing Structures

The Minimize refinement task performs a truncated-Newton energy minimization, using the OPLS2000 all-atom force field (protein-optimized) for proteins and OPLS2001 for ligands and cofactors, and treating solvation energies and effects via the Surface Generalized Born (SGB) continuum solvation model.

**To minimize all or part of the structure selected:**

1. Select Minimize from the Task menu.

   Atom selection options are displayed under the heading Residues for minimization.

2. Click All to minimize all residues, or specify a set of residues as described in Section 2.6 on page 23.

   For more information, see the online help or Chapter 5 of the *Maestro User Manual*.

3. Click the Run button to start the minimization job.

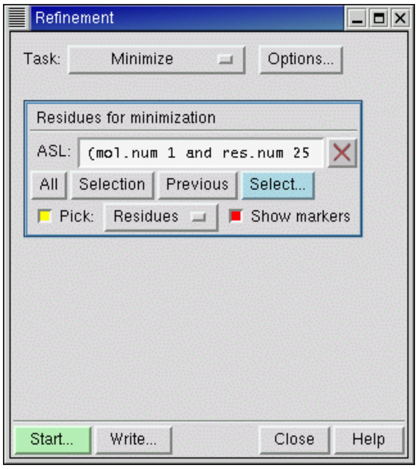
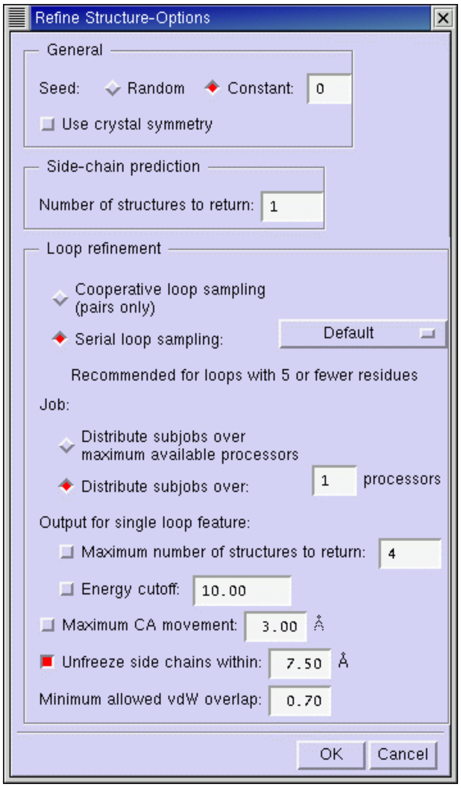


***Figure 11.5. The* Minimize *task in the* Refinement *panel.***



Refine Structure-Options

General

Seed: Random [✓ Constant:] 0

Use crystal symmetry

Side-chain prediction

Number of structures to return: 1

Loop refinement

Cooperative loop sampling (pairs only)

[✓ Serial loop sampling:] Default

Recommended for loops with 5 or fewer residues

Job:

Distribute subjobs over maximum available processors

[✓ Distribute subjobs over:] 1 processors

Output for single loop feature:

Maximum number of structures to return: 4

Energy cutoff: 10.00

Maximum CA movement: 3.00 Å

[✓ Unfreeze side chains within:] 7.50 Å

Minimum allowed vdW overlap: 0.70

***Figure 11.6. The*** Refine Structure–Options ***dialog box with*** Default ***sampling selected.***

## 11.7 Refinement Panel Options

The Options dialog box for Prime–Refinement offers three sets of options: General options, Side-Chain Prediction options, and options for Loop Refinement. These options are identical to their counterparts in the Refine Structure–Options dialog box in Figure 7.4. Click the Options button to view or change the options and settings.

### 11.7.1 General Options

The General section supplies options that apply to all types of refinement. The options include specification of a random-number seed for starting the side-chain predicition and use of crystal symmetry.

#### Seed

Specify the seed for the random-number generator used in side-chain predictions. The seed is used for side-chain prediction in both the loop refinement and the side-chain prediction tasks. The options are:

- Random: use a random seed.
- Constant: use the integer given in the text box as the seed. The default is 0.

#### Use crystal symmetry

If crystal symmetry is known for this protein, apply periodic boundary conditions so that the crystal symmetry is satisfied.

### 11.7.2 Side-Chain Prediction Options

In this section you can specify the number of structures to return from side-chain prediction. The default is 1. To return multiple structures, enter the desired number in the Number of structures to return text box.

### 11.7.3 Loop Refinement Options

The Loop Refinement section provides options for selecting the sampling method, job options, output options, and some general options.

Prime can perform cooperative loop sampling as well as serial loop sampling. Both the size of the loop and the sampling options chosen affect the time needed for loop refinement. The options are:

#### Cooperative loop sampling

Select this option to refine two loops together. Each loop is refined in turn until the refinement has converged for both loops. You must have only two loops selected to use this option.

#### Serial loop sampling

Select this option to refine each loop independently. The option menu provides five levels of sampling accuracy: Default, Extended Low, Extended Medium, Extended High, Ultra Extended. The text below the option and menu displays the recommended loop length for the selected accuracy.

Extended sampling samples loop conformations more thoroughly, using a series of loop constraint settings. Up to eight processors can be used simultaneously to perform loop refinement. Extended sampling can be run at four different sampling levels: Low, Medium, High, and Ultra.

### Output for single loop feature

These options determine how many structures are returned:

- Maximum number of structures to return: Select this option to return more than one structure for each loop specified for refinement. The default when this option is selected is to return 4 structures for each loop.
- Energy cutoff: Select this option to prevent higher-energy structures from being returned. Structures whose energy is higher than that of the lowest-energy structure by more than this amount are not returned. The default cutoff when this option is selected is 10 kcal/mol.

### Job

These options allow you to set a limit on the the number of processors over which to distribute the loop sampling subjobs. Each sampling method runs in several stages, and each stage can use a different number of processors. The maximum number of processors that can be used by the different sampling methods are:

| | |
|---|---|
| Default | 1 |
| Extended Low | 2 |
| Extended Medium | 4 |
| Extended High | 8 |
| Ultra Extended | 8*L (L is the loop length) |

- To set the maximum number of processors, select Distribute subjobs over and enter a value in the text box.
- To make use of as many processors as are available, select Distribute subjobs over maximum available processors.

The following settings are applied to the loops you have selected for refinement.

- Maximum CA movement: To limit the extent to which alpha carbon atoms can move from their original positions, select this option, and enter a limit in the text box. The default distance is 3.0 Å. The extended sampling procedure tests multiple values for Maximum CA movement, so this option is unavailable when extended sampling is selected.
- Unfreeze side chains within: Select this option to allow side chains near the specified loop to move, and specify a distance in the text box. The default distance is 7.5 Å.
- Minimum allowed vdW overlap: By default, this value is set to 0.70, meaning that the distance between atoms must be at least 70% of their 'ideal' van der Waals separation. The extended sampling procedure tests multiple values for the minimum overlap, and therefore this option is unavailable when extended sampling has been selected.

## 11.8 Incorporating Output Into the Project

By default, the structures created in Prime–Refinement are appended to the Project Table for the project currently open. If you would prefer the output structure to replace the input structure, or would rather not incorporate the output structure into the project at all, select the appropriate option from the Incorporate option menu in the Start dialog box:

- Append new entries
- Replace existing entries
- Do not incorporate

# Maestro Protein Structure Alignment

It is sometimes useful to be able to align proteins outside the usual Prime workflow. Maestro provides access to the structure alignment facilities of Prime so that you can perform such alignments, in the Protein Structure Alignment panel. A Prime license is required to run alignments from the Protein Structure Alignment panel

To open the Protein Structure Alignment panel, choose Protein Structure Alignment from the Tools menu in the main window.

In the Protein Structure Alignment panel, structure alignment is performed on Project Table entries that have been included in the Workspace. The reference structure, whose frame of reference is used for the alignment, must be the first included entry (the entry with the lowest

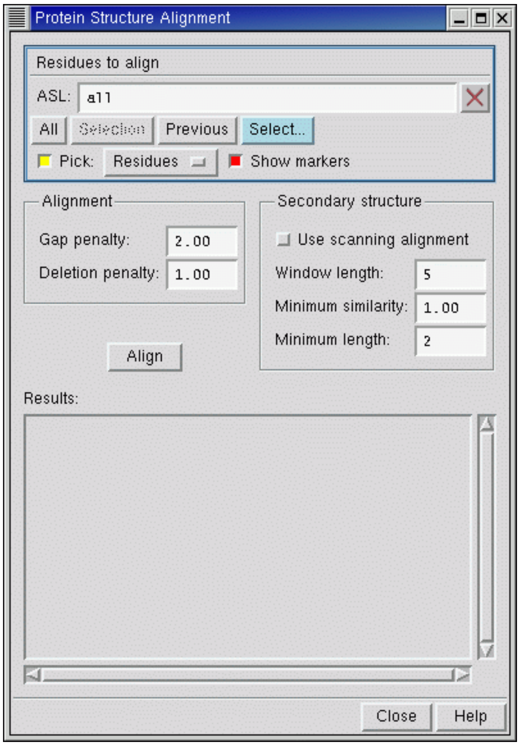


***Figure 12.1. The*** Protein Structure Alignment ***panel.***

row number). To make another protein the reference, move that protein above all the other included entries in the Project Table.

By default, the alignment includes all residues. However, it is often useful to align a subset of residues that have greater similarity than the structures as a whole, or are more informative to compare. Because this program uses matching of secondary structure elements, the subset should have enough contiguous residues to include at least one helix or strand. Very small numbers of residues do not produce useful alignments. Selection of subsets is discussed further under Residues to align, below.

When the alignment calculation is complete, the structures are placed in the same frame of reference in the Workspace, and the results of the calculation are listed in the text display area of the Protein Structure Alignment panel. The aligned residues are listed and an RMSD and Alignment Score are reported. The RMSD is calculated based only on those residues that are considered to have been successfully aligned, and therefore does not represent an overall comparison of the structures. An Alignment Score greater than about 0.7, or a failure of the structural alignment calculation, indicates there is insufficient structural similarity for a meaningful alignment.

To run a default alignment job, include two or more protein structures in the Workspace, open the Protein Structure Alignment panel, and click Align to start the alignment calculation.

The Protein Structure Alignment panel provides some tools for controlling which parts of the protein are aligned and how they are aligned. These tools are described below.

### Gap penalty

In the structure alignment calculation, this is the alignment score penalty for initiating a gap in the alignment. The default gap penalty is 2.00. The main purpose of this parameter is to prevent the generation of very poor initial alignments.

### Deletion penalty

In the structure alignment calculation, this is the penalty for extending a gap. The default deletion penalty is 1.00.

### Use scanning alignment

This option is deselected by default. When it is selected, structure alignment is carried out in scanning rather than the default global mode. Generally, the default global mode is more useful. However, if you do not find any similarity between structures with global alignment, you can try scanning alignment.

Global alignment involves running double dynamic programming on the full scoring matrix built from all secondary structural elements (SSEs) in both structures. In scanning alignment double dynamic programming is run on successively smaller subregions of the full scoring matrix (window sizes of 20, 15, 10 and 5 SSEs) until a reasonable alignment is produced. Assuming a reasonable alignment is found it is used as a starting point for building a global alignment (involving all SSEs).

### Window length

Number of successive SSEs that are to be used as a baseline for the global alignment. The default is 5. This option is significant only if Use scanning alignment has been selected.

### Minimum similarity

This is the alignment score above which alignments are reported as unsuccessful. The default is 1.00, where alignment scores of 0.7 or less mean greater similarity, and alignment scores above 1.0 mean less similarity. When alignment scores are greater than 1.0, the alignment is unlikely to be meaningful.

Structural similarity of proteins is measured by the Protein Structural Distance (PSD). This measure combines the similarity scores for the secondary structural elements obtained from dynamic programming and the RMSD of aligned residues for the optimal alignment. For details see Yang, A.; Honig, B. *Journal of Molecular Biology* **2000**, *301*, 665-678.

### Minimum length

Minimum number of SSEs that are to be aligned in scanning alignment. The default is 2.

### Residues to align

Alignment can be performed for all residues or for a subset of residues. You might need to try different subsets to obtain the most useful alignment. You can use the picking controls, Workspace selection, or the Atom Selection dialog box to specify subsets. Subsets should include at least one secondary structure element (helix or strand). Clicking the Select button opens the Atom Selection dialog box to the Residue folder. Here you can select a range of residue numbers, specify portions of the residue sequence, or click Secondary Structure in the list of options on the left to choose residues based on their location in Helix, Strand, or Loop structures. Subsets can be combined, intersected, or otherwise modified in the Atom Selection dialog box.

# Command Syntax

This chapter summarizes the command syntax for various programs that are part of Prime. All the programs are executed by driver scripts, which run under the Schrödinger Job Control facility. For more information on this facility, see the *Job Control Guide*. The conventions used in describing the command syntax are outlined in the Document Conventions section at the front of this manual.

Residue specifications are used with a number of the input file keywords for several of the scripts. These have the format *C*:*N*, where *C* is a one-character chain ID ('_' if the chain ID is blank) and *N* is the residue number and insertion code (if any). Some of the programs prefix the chain ID with `CHAIN` when they write out files, e.g. `CHAINA:40` instead of `A:40`. This prefix is ignored in subsequent reading of the residue specification.

Most of these scripts accept the standard Schrödinger Job Control command line options and other commonly used options, listed in Table 13.1. Exceptions are noted in the program descriptions.

*Table 13.1. Standard Schrödinger job control options.*

| Option | Description |
|---|---|
| `-HOST` *host* | run the job on the specified host. The default is the local host. |
| `-LOCAL` | Run the job in the local directory rather than the temporary directory |
| `-TMPDIR` *tmpdir* | Run the job in the temporary directory *tmpdir* |
| `-WAIT` | Do no return control to the shell until the job finishes |
| `-INTERVAL` *n* | Update the output every *n* seconds |
| `-NICE` | Run the job at reduced priority |
| `-DEBUG` | Turn on debugging for job control as well as the program |
| `-HELP` | Show information on command-line usage |

## 13.1 blast—Run Blast Searches

The `blast` script provides an interface to the Blast and PSIBlast programs from NCBI, while also supporting standard Schrodinger Job Control options. All options are passed on to the script via an input file.

## Syntax

```
$SCHRODINGER/blast jobname [options]
```

Input is read from the file *jobname*`.inp`. Output is written by default to *jobname*`.out`.

## Options

The blast script accepts the options listed in Table 13.1.

## Input File

Keywords control the operation of the `blast` script, and are listed in Table 13.2. Most keywords are simply a translation of options available from the `blastall` or `blastpgp` executables distributed by NCBI. For optional keywords, the Blast default values are used. Since these defaults can differ between different version of Blast they are not listed here.

*Table 13.2. Keywords for the blast script*

| Keyword | Description |
|---|---|
| `QUERY_FILE` *filename* | Required. Name of the FASTA file containing the input/query sequence |
| `DATABASE {nr|pdb}` | Database to be used in the search. In a default installation the two allowed values are `nr` and `pdb`. |
| `ALIGNMENT_FORMAT` *value* | Alignment view options (Blast `-m` option). Allowed values are:<br>`0` pairwise<br>`1` query-anchored showing identities<br>`2` query-anchored no identities<br>`3` flat query-anchored, show identities<br>`4` flat query-anchored, no identities<br>`5` query-anchored no identities and blunt ends<br>`6` flat query-anchored, no identities and blunt ends<br>`7` XML Blast output<br>`8` tabular<br>`9` tabular with comment lines<br>`m2io` Maestro format (default) |
| `EXPECT` *value* | Expectation value (E) (Blast `-e` option). |
| `FILTER {true|false}` | Filter query sequence: DUST with `blastn`, SEG with others. (Blast `-F` option). |
| `GAP_OPEN_COST` *value* | Cost to open a gap (Blast `-G` option) |
| `GAP_EXTEND_COST` *value* | Cost to extend a gap (Blast `-E` option) |

*Table 13.2. Keywords for the blast script (Continued)*

| Keyword | Description |
|---|---|
| `MATRIX` *type* | Sequence similiarity matrix to use for the alignments. Allowed values of *type* are `BLOSUM45`, `BLOSUM62`, `BLOSUM80`, `PAM30`, `PAM70` |
| `MAX_ROUNDS` *n* | Maximum number of passes to use in multipass version (PSI-Blast `-j` option); only applies to runs using `PROGRAM blastpgp` |
| `E_VALUE_THRESHOLD` *value* | e-value threshold for inclusion in multipass model (PSIBlast `-h` option). |
| `OUTPUT_FILE` *filename* | Name of the output file. Default: *jobname*`.out`. |
| `PROGNAME` *name* | Type of blast search to run. The allowed values are `blastp`, `blastpgp`, and `bl2seq` (for pairwise alignments). Default: `blastp`. |
| `TEMPLATE_FILE` *filename* | Name of the FASTA file that contains the template sequence (`bl2seq -j` option). Only applies to pairwise alignments using `bl2seq` |
| `PSSM_ASCII_OUTFILE` *filename* | ASCII output file for PSI-BLAST matrix (PSIBlast `-Q` option). |
| `EXPAND_HITS {true\|false}` | Expand redundant hits, based on information in the NCBI defline for a particular hit. Only available when using Maestro format. |
| `SHOW {all\|pdb}` | Allows the user to restrict the types of hits that are included in the output. `all` includes all hits; `pdb` only includes hits from the PDB. Only available when using Maestro format. |

## Return Value and Errors

`blast` returns 0 if successful and a non-zero value in all other cases.

## Examples

The following is an example input file; `query.aa` is the query sequence in FASTA format.

```
QUERY_FILE         query.aa
PROGNAME           blastp
DATABASE           pdb
```

## Environment Variables

The following environment variables are supported by the blast script:

| | |
|---|---|
| `PSP_BLASTDB` | directory containing the customized Blast databases |
| `PSP_BLAST_DIR` | directory containing a custom Blast installation |

## 13.2 bldstruct—Build a Protein Model from an Alignment

The `bldstruct` script can be used to build initial (unrefined) protein models from an alignment to one or more templates via the Protein Local Optimization Program (PLOP). `bldstruct` handles all the tasks associated with the building of homology models via PLOP (i.e. parsing and preparation of input files, interaction with PLOP, and handling of output structures, etc). All jobs handled by `bldstruct` are run under Schrödinger Job Control.

### Syntax

```
$SCHRODINGER/bldstruct jobname [options]
```

### Options

The `-DEBUG` and `-HOST` options are recognized by `buildbb`. See Section 4.3 of the *Job Control Guide* for more information.

### Command Input File

The command input file is named *jobname*`.inp`. This input file contains keyword-value pairs, one pair per line separated by one or more spaces. The keywords are given in Table 13.3.

*Table 13.3. Keywords for the bldstruct script*

| Keyword syntax | Description |
|---|---|
| `TEMPLATE_NAME` *template* | Required. A label used to identify a given template in subsequent data fields. For example, if `TEMPLATE_NAME` is set to `1ABC`, other template-specific fields are given as `1ABC_STRUCT_FILE`, `1ABC_ALIGN_FILE`, and so on. The chain to be used is specified by an underscore and letter suffix on the name: for example, for `1BPJ_A`, chain A will be used. |
| *template*`_NUMBER` *n* | Required. Used to identify a given template in the `COMPOSITE_ARRAY` (see below). Numbering goes from 1 to 9. |
| *template*`_STRUCT_FILE` *filename* | Required. The PDB file to be used for this particular template. |
| *template*`_ALIGN_FILE` *filename* | Required. The name of the file containing the alignment between this template and the query. Details on the format are provided in the Examples section below. |

*Table 13.3. Keywords for the bldstruct script (Continued)*

| Keyword syntax | Description |
|---|---|
| *template*_HETERO_*i* *ligand-spec* | Required. Specifies a ligand to include from this template. The format of the specification is<br>`AAA C:###`<br>where `AAA` is the ligand's three letter code, `C` is its chain ID, `###` is its residue number. Multiple ligands from the template are numbered using *i* as an index, starting from zero. |
| `TAILS` *n* | Specifies how much of the terminal tails to construct. The available options are:<br>0 omit tails entirely<br>1 build to outermost aligned residue<br>2 build complete tails<br>Default: 1. |
| `SIDE_OPT {true\|false}` | Turn on or off a side chain optimization stage. Default: `true`. |
| `MINIMIZE {true\|false}` | Turn on or off minimization of regions that were not directly copied from the template structure. These regions primarily include portions of the structure involved in closing gaps or building insertions, but also include any side chains optimized due to the `SIDE_OPT` keyword. Default: `true`. |
| `KEEP_ROTAMERS {true\|false}` | Specifies which side chains to optimize. If `true`, the rotamers for conserved side chains are preserved, and only non-conserved side chains are predicted. If `false`, all side chains are predicted. This keyword is ignored if `SIDE_OPT` is `false`. Default: `true`. |
| `COMPOSITE_ARRAY` *string* | A string of integers indicating from which template (or, more specifically, from which alignment) coordinates for a given residue should be obtained. Thus, this string of integers must be equal in length to the query sequence being used in the alignment files. `COMPOSITE_ARRAY` is required for multiple template jobs, but is ignored for single template jobs. |
| `MAX_INSERTION_SIZE` *n* | Specifies the longest insertion in the alignment that will be built. Insertions longer than this will be omitted, and not appear in the output structure. Residue numbering will reflect the full query sequence however. Default: 1000. |

*Table 13.3. Keywords for the bldstruct script (Continued)*

| Keyword syntax | Description |
|---|---|
| `BUILD_DELETIONS {true\|false}` | Turn on or off closure of the chain breaks near deletions in the alignment. If not closed, they will remain as chain breaks in the output structure. Default: `true`. |
| `BUILD_TRANSITIONS {true\|false}` | Turn on or off closure of the junctions between templates in multi-template homology modelling jobs. If not closed, they will remain as chain breaks in the output structure. Default: `true`. |
| `USE_SYMMETRY {true\|false}` | Turn on or off use of the crystallographic symmetry of the template file when constructing the homology model. Default: `false`. |

## Files

In addition to the input files (structure and alignment) and the job files (command input file, *jobname*`.inp`, and log file, *jobname*`.log`), `bldstruct` produces the following output files:

| | |
|---|---|
| *jobname*`-out.mae` | Built model in Maestro format. |
| *jobname*`-out.pdb` | Built model in PDB format. |
| *jobname*`.psane` | ProSane diagnostic information. |

## Return Value and Errors

`bldstruct` returns a value of 0 on successful completion, 1 otherwise. All error messages are printed to the file *jobname*.log. There is no comprehensive listing of possible errors as these can arise in a variety of programs and scripts.

## Examples

The following is a sample input file for building on two templates:

```
QUERY_OFFSET        0
1TSV_NUMBER         1
1TSV_STRUCT_FILE    pdb1TSV.ent
1TSV_ALIGN_FILE     1TSV.align
TEMPLATE_NAME       1BPJ_A
1BPJ_A_NUMBER       2
1BPJ_A_STRUCT_FILE  pdb1BPJ_A.ent
1BPJ_A_ALIGN_FILE   1BPJ_A.align
1BPJ_A_HETERO_0     UMP :317
1BPJ_A_HETERO_1     ZN :35
TAILS               0
SIDE_OPT            true
MINIMIZE            true
KEEP_ROTAMERS       true
```

```
MAX_INSERTION_SIZE 1000
BUILD_DELETIONS    true
BUILD_TRANSITIONS  true
USE_SYMMETRY       false
COMPOSITE_ARRAY 111111111111111111111122222222222222222222222222222222222221111111
```

The alignment file format is as follows. The first line corresponds to the query sequence (indicated by `ProbeAA:`) and the second line corresponds to the template sequence (indicated by `Fold AA:`). A period is used as a gap character, and an `X` is used for non-standard residues. Below is a sample alignment file.

```
ProbeAA: .AAEEKTEFDVILKAAGANKVAVIKAVRGATGLGLKEAKDLVESA...PAALKEGVSKDDAEALKKALEEAG
AEVEVK
Fold AA: AAQEEKTEFDVVLKSFGQNKIQVIKVVREITGLGLKEAKDLVEKAGSPDAVIKSGVSKEEAEEIKKKLEEAG
AEVELK
```

The sequences are split for formatting reasons, and should be on a single line in the actual file.

## 13.3 buildbb—Build Backbone

The Build Backbone program in Prime constructs three-dimensional structures of the query sequence from templates selected by the Fold Recognition step. Because of the low-sequence identity nature of the cases this program was designed for, the structures it builds are not only based on a given template, but it also intelligently selects fragments of proteins that are structurally related to the template, then uses tertiary folding potential to optimize the composite structure.

There are several steps in Build Backbone:

- **Constructing Composite Templates and Alignments**—Composite templates are constructed by first identifying the structurally conserved and the variable regions of a seed template and its structural neighbors, then enumerating representative subsequences of each variable region before merging with the conserved subsequences taken from the seed template. As a result, a set of composite templates can be generated from one seed template and its structural neighbors. Each of these composite templates is aligned to the qeury sequence with a scoring function that includes profile-sequence and secondary structure matching.
- **Generating Tertiary Folding Constraints**—Fragments taken from different proteins in each composite templates were mapped and rotated onto the same coordinate frame based on multiple structure alignment of the proteins. Constraints are derived from alignment for each composite template, upon which a model of the query structure is built. In this step, the number of composite templates is reduced by clustering and comparing the representatives in each cluster. Only unique and plausible structures are saved for tertiary

folding.

- **Tertiary Folding and Simulation**—Constraints derived from alignments are passed on to tertiary folding simulation, which also supplies the appropriate run-time parameters based on predetermined semi-empirical scoring functions. Unaligned residues in the query sequence are optimally added to the final structure.
- **Global Scoring**—In this step, structures produced from composite templates based on each corresponding seed template are evaluated by the global scoring function, which not only includes tertiary folding potential terms but also fold recognition scores from FR step. The top 5 structures, and in most cases the top structure, are the best backbone models built for the query.

## Syntax

```
$SCHRODINGER/buildbb jobname [options]
```

## Options

The `-DEBUG` and `-HOST` options are recognized by `buildbb`. See Section 4.3 of the *Job Control Guide* for more information.

## Command Input File

The command input file is named *jobname*`.inp`. This input file contains keyword-value pairs, one pair per line separated by one or more spaces. The keywords are given in Table 13.5.

*Table 13.4. Keywords for the buildbb script*

| Keyword syntax | Description |
|---|---|
| `QUERY_NAME` *name* | Name of the query sequence |
| `QUERY_FILE` *filename* | Query sequence file, in FASTA format |
| `ALIFORMAT` *value* | Select format of alignments file. Default is Maestro format. For plain text format set *value* to `dev`. |
| `BGNRES` *n* | The first residue of the query that is used in search for homologs. Default: 1 (the first residue). |
| `ENDRES` *n* | The last residue of the query that is used. Default: the last residue of the complete query |

*Table 13.4. Keywords for the buildbb script (Continued)*

| Keyword syntax | Description |
|---|---|
| `PREDSS[_i]` *filename* | Secondary structure prediction file of the query sequence, in CASP format. One occurrence of this keyword is required for each file, and there must be at least one SSP file specified. If only one prediction is used, the keyword is PREDSS; if more predictions are used, the keyword is suffixed with an index, starting from zero: `PREDSS_0, PREDSS_1`, and so on. |
| `FR_FILE` *filename* | Complete file name (with absolute path) of the Fold Recognition output ranking file (`.out`). Must be obtained from Maestro or from a run of `fr`. |
| `SSP_FILE` *filename* | Complete file name (with absolute path) of the secondary structure prediction file, in Maestro format. Must be obtained from Maestro. |
| `TEMPLATE_NAME` *name* | Seed templates that are selected to build structure on. If run inside Maestro, the top 25 templates in the Fold Recognition step are automatically passed on to Build Backbone. The minimum number of templates is 1. |
| `CLUSTERING` *n* | Level of clustering of initial models built directly from composite templates and alignments. Default: 2 (medium and recommended); Other values: 1(low clustering) and 3 (high clustering). |
| `FILTERING` *n* | Level of filtering of initial models. Some structures are filtered out because they are displaced by other, better structures. Default: 2 (medium and recommended); Other values: 1 (low filtering, more structures) and 3 (high filtering, fewer structures). |
| `MAX_NTEMP` *n* | Always set to 5000. |
| `TOP_COMPOSITES` *n* | Always set to 1. |
| `TOP_FAMILIES` *n* | Always set to 10. |
| `MAX_STRUC_RET` *n* | Maximum number of structures returned for each seed template. Default and recommended value: 20. |
| `HOST` *hostname* | Host name as specified in the `schrodinger.hosts` file. |
| `NSUBJOB` *n* | The maximum number of templates that are submitted simultaneously. This number should be less than or equal to the total number of tokens available or the total number of processors available, whichever is smaller, on a non-queuing system. On a queuing system, this number should be less than or equal to the total number of templates specified by `TEMPLATE_NAME`. |

## Files

- Input file—named *jobname*`.inp`. Contains keywords for running the program.
- Query sequence file—File containing the complete sequence of the query, in FASTA format. Specified in the input file.

- Secondary structure prediction files in CASP format for the query sequence. See below for an example of CASP format. Specified in the input file.
- Fold Recognition Output Ranking File—File named *jobname*`.out` containing a list of templates ranked according to the Fold Recognition scoring function. Specified in the input file. Templates selected in the input file above can be a subset of the templates listed in this file. This file is only used in global scoring.
- Secondary structure prediction file in Maestro format—This file contains all the secondary structure predictions used for the query sequence, and is only used in global scoring. Specified in the input file.
- Output tar file—File named *jobname*`.out.tar` containing the results. Final structures and intermediate structures (i.e. with gaps and unaligned residues) from each seed template, together with corresponding alignments, are saved in one directory. The directories for each seed template are then archived into one tar file.
- Log file—File named *jobname*`.log` containing progress of the program, including Job Control messages for each subjob (template), warnings, and error messages.

## Return Value and Errors

The script returns 0 for success and 1 for failure.

In case of failure, check the log file for details on what might be have caused the program to fail. To turn on debugging output use the `-DEBUG` command line option.

## Examples

The following is an example input file, `1hl6A.inp`, which will submit three templates simultaneously to the host `cluster`, a cluster with a queuing system. When all three templates finish, the output tar file `1hl6A.out.tar`, which contains the output files, and the log file `1hl6A.log` should be returned.

```
QUERY_NAME      1HL6:A
QUERY_FILE      prime_build_bb_run1_1hl6A-query.seq
HOST    cluster
FR_FILE         /home/username/prime_build_bb_run1_1hl6A-
foldrec.out
SSP_FILE        /home/username/prime_build_bb_run1_1hl6A-ssp.seq
PREDSS_0        prime_build_bb_run1_1hl6A-ssp1.casp
PREDSS_1        prime_build_bb_run1_1hl6A-ssp2.casp
PREDSS_2        prime_build_bb_run1_1hl6A-ssp3.casp
TEMPLATE_NAME   d1b7fa1
TEMPLATE_NAME   d1b63a_
```

```
    TEMPLATE_NAME   d2scua_
    BGNRES  1
    ENDRES  165
    CLUSTERING      2
    FILTERING       2
    MAX_NTEMP       5000
    TOP_COMPOSITES  1
    TOP_FAMILIES    10
    MAX_STRUC_RET   20
    NSUBJOB 3
```

Below is an example of the plain text format for output alignment files. This format is generated when you include `ALIFORMAT dev` in the command input file.

```
probe length=70======tmplt_T0281_d1dq3a2 length=79 TotalScore= 42 SEQ= -4 SSTR= 102
 SEQID= 0.061538 nali= 65
probe bgnRes=1 endRes=70    template bgnRes=3 endRes=75

ProbeAA: ..MWMPPRPEEVARKLRRLGFVERMAKGGHRLYTHPDGRIVVVPFH...SGELP....KG
ProbeSS:   LLLLLLHHHHHHHHHHLLLEEEEELLLEEEEELLLLLEEEeLLL   LLLLL    HH
Fold AA: GNFGLPLNFNAFKEWASEYGVEFKTNGSQTIAIIND...ERISLGQWHTRNRVSKAVLVK
Fold SS: LLLEELLLHHHHHHHHHLLLLEEEEELLEEEEEELL   EEEELLLHHHHLLEEHHHHHH

ProbeAA: TFKRILRDAGLTEEEFHNL....
ProbeSS: HHHHHHHHhLLLHHHHHLL
Fold AA: MLRKLYEATK.DEEVKRMLHLIE
Fold SS: HHHHHHHHHL LHHHHHHHHHHL
```

The file contains header information at the top, then in the alignment section, the query sequence (`ProbeAA`), query composite secondary structure (`ProbeSS`), template sequence (`Fold AA`) and template secondary structure (`Fold SS`) are given. Gaps are denoted by periods in the sequences and by spaces in the secondary structures.

## 13.4 fr—Fold Recognition

`fr` finds template proteins that are similar to the query sequence at both sequence and structure levels.

The fold recognition program used in Prime differs from standard sequence search program such as BLAST by taking into account secondary structure matching and profile-sequence matching. As a result, `fr` can find structural homologs with low sequence identity to the query (<15-20%), which makes it a desirable tool when a given query does not have any high sequence identity homologs in PDB.

Templates used by `fr` are stored in Prime's internal fold library, which is a non-redundant subset of the PDB database and is updated periodically. Individual domains of each multi-domain template are also included in the fold library. When selecting homologs for a query sequence, a scoring function takes into account profile-sequence matching, secondary structure matching and length compatibility between the query sequence and a template, for either a full chain entry or a domain entry. Templates are ranked based on this score.

In this release, we have improved the accuracy of fold recognition with a new set of PSI-BLAST parameters for more accurate profile-sequence matching and with increasing accuracy in secondary structure predictions. As in the previous releases, the top 25 templates are automatically selected in Maestro after Fold Recognition for the next step in the Threading path, Build Backbone. However, because of the enhancements, we now recommend selecting only the top 20 templates if the query is mainly beta according to the secondary structure prediction. For all other secondary structure types, select the top 10 if the query sequence is short (<=120 residues) or the top 5 if the query sequence is longer.

Please note that at least one secondary structure prediction of the query sequence has to be provided in order to run `fr`. `fr` does not predict secondary structure itself.

## Syntax

`$SCHRODINGER/fr` *jobname* `[`*options*`]`

## Options

The `-DEBUG` and `-HOST` options are recognized by `fr`. See Section 4.3 of the *Job Control Guide* for more information.

## Command Input File

The command input file is named *jobname*`.inp`. This input file contains keyword-value pairs, one pair per line separated by one or more spaces. The keywords are given in Table 13.5.

*Table 13.5. Keywords for the fr script.*

| Keyword syntax | Description |
|---|---|
| `QUERY_NAME` *name* | Name of the query sequence |
| `QUERY_FILE` *filename* | Query sequence file, in FASTA format |
| `ALIFORMAT` *value* | Select format of alignments file. Default is Maestro format. For "normal" format set *value* to `dev`. |

*Table 13.5. Keywords for the fr script.*

| Keyword syntax | Description |
|---|---|
| `BGNRES` *n* | The first residue of the query that is used in search for homologs. Default: 1 (the first residue). |
| `ENDRES` *n* | The last residue of the query that is used. Default: the last residue of the complete query |
| `PREDSS`[_*i*] *filename* | Secondary structure prediction file of the query sequence, in CASP format. One occurrence of this keyword is required for each file, and there must be at least one SSP. If only one prediction is used, the keyword is PREDSS; if more predictions are used, the keyword is suffixed with an index, starting from zero: `PREDSS_0`, `PREDSS_1`, and so on. |

## Files

- Input file—named *jobname*`.inp`. Contains keywords for running the program.
- Query sequence file—File containing the complete sequence of the query, in FASTA format. Specified in the input file.
- Secondary structure prediction files in CASP format for the query sequence. See below for an example of CASP format.
- Output Ranking File—File named *jobname*`.out` containing a list of templates ranked according to the FR scoring function as described above.
- Output alignment file—File named *jobname*`.ali` containing pairwise alignment between the query and a template. By default, the alignment is in Maestro format (m2io). In order to view alignments in normal format (i.e. aligned residues from query and template are placed on top of each other), add the keyword `ALIFORMAT` to the input file with value set to `dev`.
- Log file—File named *jobname*`.log` containing progress of the program, including warnings and error messages.

## Return Value and Errors

The script returns 0 for success and 1 for failure.

In case of failure, check the log file for details on what might be have caused the program to fail. To turn on debugging output use the `-DEBUG` command line option as follows:

```
$SCHRODINGER/fr -DEBUG jobname
```

### Examples

The following file demonstrates the use of three SSPs.

```
QUERY_NAME      HypProtein_furiosus_truncated
QUERY_FILE      HP.seq
PREDSS_0        prime_foldrecog_run1_HP-ssp1.casp
PREDSS_1        prime_foldrecog_run1_HP-ssp2.casp
PREDSS_2        prime_foldrecog_run1_HP-ssp3.casp
BGNRES          1
ENDRES          197
```

The following is an example of a CASP format file. The CASP format has a minimum of 2 columns per line, with the first being the one-letter code for the amino acid and the second being the secondary structure type. An optional third column gives the confidence level of the prediction.

```
PFRMAT SS
TARGET 1HL6:A
AUTHOR xxxx-xxxx-xxxx
REMARK written by Schrodinger LLC Software
METHOD not applicable
MODEL 1
M C 1.00
A C 1.00
D H 1.00
V H 1.00
...
E C 1.00
K C 1.00
R C 1.00
R C 1.00
R C 1.00
END
```

## 13.5 multirefine—Multi-Step Extended Sampling

The multirefine script automates extended sampling protocols, which consist of multiple executions of `refinestruct` on a set of structures. Multirefine handles the submission of the subjobs and the compilation of the results. Currently available protocols include extended loop sampling, ultra-extended loop sampling, and cooperative loop pair sampling.

## Syntax

```
$SCHRODINGER/multirefine input-file [options]
```

where *input-file* is the name of the command input file, with or without a `.inp` extension, and serves also as the default job name. The input file must be either the first or last command-line argument, and everything else on the command line is passed as is to Job Control.

## Options

The standard Job Control command line options listed in Table 13.1 are accepted. There are no options specific to this program.

## Command Input File

Most of the keywords are the same as the general keywords and the keywords for protocol 1 (loop and helix refinement) for `refinestruct`: see Section 13.7 on page 145 for details. The differences and new keywords are described below.

*Table 13.6. Keywords for the multirefine input file*

| Keyword syntax | Description |
|---|---|
| `QUERY_FILE` | Required. The input structure file in Maestro format. Same as for `refinestruct`. |
| `LOOP_`*i*`_RES_`*j* | Specify beginning and end of loops. Same as for `refinestruct`. Required if `SEG_LIST` is not given. Depending on the protocol, 1, 2, or an arbitrary number of loops may be specified. For example, to specify the beginning and end of loop 0, from residue 10 to residue 20 in chain A, use the following:<br>`LOOP_0_RES_0 A:10`<br>`LOOP_0_RES_1 A:20` |

Table 13.6. Keywords for the multirefine input file (Continued)

| Keyword syntax | Description |
|---|---|
| `PROTOCOL` *n* | Type of refinement to be carried out. Can be specified as either a number or text string. Different from `refinestruct`. Available choices are:<br>`0` or `default`: Equivalent to running a single loop refinement using `refinestruct`. Not particularly useful except for completeness and testing purposes. Does not recognize `THREADS` keyword.<br>`1` or `extended`: Used for Extended Sampling from the GUI. Does not recognize `MIN_OVERLAP` and `MAX_CA_MOVEMENT` keywords, as these are used internally by the protocol. Accepts multiple loops as input which will be run sequentially.<br>`2` or `long_loop`: Used for Ultra-extended Sampling from the GUI. Does not recognize `MIN_OVERLAP` and `MAX_CA_MOVEMENT` keywords, as these are used internally by the protocol. Also does not recognize `THREADS` keyword, as the sampling is determined by the loop length. Only one loop can be specified.<br>`3` or `loop_pair`: Used for Cooperative Loop Sampling from the GUI. Exactly two loops must be specified. |
| `SEG_LIST` *filename* | Name of a file containing the list of loops to be refined. For the example given for the keyword `LOOP_`*i*`_RES_`*j*, the file would consist of the single line:<br>`loop A:10 A:20`<br>Not available with `refinestruct`. |
| `THREADS` *n* | Number of simultaneous jobs to be run during selected refinement stages. Determines the sampling level for some protocols. Note: the jobs specified by `THREADS` do not have to actually run in parallel—see `MAX_JOBS`. Not available with `refinestruct`. |
| `MAX_JOBS` *n* | Maximum number of subjobs that can run at one time. This can be set to a value less than the value of `THREADS` if sufficient resources are not available. Not available with `refinestruct`. |
| `HOST` *hostname* | Job host to which the subjobs will be submitted. The top-level driver script will run on the host specified by the `-HOST` command-line argument (or on the launch host if none is specified.) If *hostname* is the same as that specified by `-HOST` or is `localhost`, the subjobs will run on the same host as the top-level driver (or be submitted to the same queuing system if the top-level host is a queue). Otherwise, the driver will submit subjobs using *hostname* as their command-line argument. Not available with `refinestruct`. |

## Files

In addition to the *jobname*.log and *jobname*.err files, `multirefine` produces the following output files:

| | |
|---|---|
| *jobname*`-out.mae` | Final structures, all in a single file. |
| *jobname*`.psane` | ProSane diagnostic output for the lowest-energy structure. Other structures have corresponding files *jobname*`-2.psane`, and so on. |
| *jobname*`.pdb` | PDB format file corresponding to lowest-energy structure. Other structures will have corresponding files *jobname*`-2.pdb`, and so on. |

Temporary files, which are normally deleted when no longer needed unless the debugging option is used, are named according to the following patterns:

| | |
|---|---|
| *jobname*`-tmp-`*N*`.*` | Files associated with temporary structures. All active structures in the ensemble have a unique serial number *N*. |
| *jobname*`-run-`*N*`.*` | Files associated with a given subjob. All subjobs have a unique serial number *N*. |

## Return Value and Errors

`multirefine` returns a value of 0 on successful completion, 1 otherwise. All scripts run by `multirefine` that return a value of 1 also print a message to the file *jobname*`.err`. This file is empty otherwise, as all non-fatal messages are printed to the file *jobname*`.log`. Output from Job Control goes to standard output, and any errors arising before the driver script has started likewise go to standard output or standard error.

## Examples

The following is an example of an input file produced by Maestro for a Serial loop sampling (Extended Medium) job:

```
QUERY_FILE          prime_refine.mae
PROTOCOL            1
LOOP_0_RES_0        CHAIN_:35
LOOP_0_RES_1        CHAIN_:40
THREADS             4
MAX_JOBS            2
NUM_OUTPUT_STRUCT   4
RES_SPHERE          7.50
USE_CRYSTAL_SYMMETRY        false
USE_RANDOM_SEED     true
SEED                0
```

```
    HOST                localhost
```

Setting PROTOCOL to 1 selects the extended sampling protocol; setting THREADS to 4 corresponds to the Medium sampling level. Setting MAX_JOBS to 2 requires that no more than 2 subjobs can run at once (e.g. if using a 2-processor machine.)

### Environment Variables

In addition to the standard Schrödinger environment variables , multirefine recognizes the following environment variables:

| | |
|---|---|
| PSP_RB_DEBUG<br>PSP_DEBUG | If this environment variable is set, debugging will be turned on and the subdirectory containing intermediate files will be kept. Equivalent to using the command-line option -DEBUG. |
| PSP_RB_LOCAL<br>PSP_LOCAL | If this is set, all calculations will run in the local launch directory. Equivalent to using the command-line option -LOCAL. |
| TFM_QUIET<br>TFM_VERBOSE | These environment variables can be set to decrease or increase the job-related output in the log file. |

## 13.6 refine—Backbone Refinement Module

Refine automatically sets up and carries out one of several predetermined series of simulations. This includes extracting input and constraint information from a composite template structure, supplying the appropriate run-time parameters for the simulation, parsing the simulation output, and if necessary carrying out additional simulations as part of a multi-step procedure. Refine also evaluates the global scoring function for the final output structures for some protocols.

### Syntax

`$SCHRODINGER/refine` *input-file* [*options*]

where *input-file* is typically the job name, with or without a .inp extension. The input file must be either the first or last command-line argument, and everything else on the command line is passed as is to Job Control.

### Options

The standard Job Control command line options listed in Table 13.1 are accepted. There are no options specific to this program.

## Command Input File

The input file specified on the command line consists of keyword-value pairs, one pair per line separated by one or more spaces. These pairs correspond to the global options used by the driver script as well as some user-defined parameters used by individual protocols. All options not specified as required have default values and can be safely omitted. Unknown keywords are ignored or passed on.

The keywords are listed in Table 13.7.

*Table 13.7. Keywords for the refine input file*

| Keyword syntax | Description |
|---|---|
| **General options** | |
| `PROTOCOL` *string* | Required Specifies the refinement protocol to be run. The possible choices are given in Table 13.8. |
| `JOBNAME` *string* | Specify the job name used by Job Control. The default is the name of the input file without the `.inp` extension. Setting `JOBNAME` to `default` explicitly generates a unique name based on the value of `STRUCT_NAME`. |
| `INPUT_DIR` *directory* | Directory to look for input files (other than the command input file) if different from launch directory. |
| `BATCH_HOST` *hostname* | Host for submitting subjobs in parallel (if applicable). Default is the host on which the driver script is running. |
| `BATCH_USER` *username* | User name for submitting subjobs in parallel (if applicable). Default is user name under which the driver script is running. |
| `BATCH_SUBMIT` *n* | Set equal to 1 if user and host keywords allow multiple jobs to be submitted simultaneously. Set equal to 0 (default) if only one processor is to be used. This option is ignored unless the selected protocol specifically allows for simultaneous simulations. |
| `STRUCT_NAME` *structname* | Root part of input structure file names *stuctname*`.pdb` if automatically generated names are used. This is an alternative to `TEMPLATE_FILE_N`. |
| `RESULT_TABLE` *filename* | Name of result table file. Overrides the default, *jobname*`.tab`. |
| `{ALIGNMENT_FILE|ALIGN_FILE}` *filename* | Name of file containing composite template alignment. Default is *structname*`.align` (see `STRUCT_NAME`). |
| **Template input options** | |
| `CHAIN_ID` *chain* | Specify the chain to be used in the case where the template file is a multi-chain PDB file. The default value of `0` means no chain IDs. |

*Table 13.7. Keywords for the refine input file (Continued)*

| Keyword syntax | Description |
|---|---|
| `USE_HB_SS` *n* | Setting this option to 1 allows the redefinition of the secondary structure based on the template H-bond search, if it differs from that specified in the `SHEET`/`HELIX` records or the predicted secondary structure. Note: this can produce an error if an attempt is made to reassign helix to sheet. The default is 0. |
| `TEMP_DOMAIN` *n* | If multiple templates are specified, setting this option to 1 will automatically treat each template as a separate domain in the simulation. The default is 1, but this option should be set to 0 if the templates contain residues in common as this would lead to an invalid domain assignment. |
| `TEMPLATE_FILE_`*N* *filename* | Name of PDB file containing composite template. Multiple occurrences of this keyword with *N* = 0, 1, 2, ... can be used if multiple homologs are aligned to the target. |
| `TEMPLATE_NAME_`*N* *name* | Template name consisting of the name of the homolog structure in the fold library and the serial number of the alignment. |
| `SAFE_DMAT` | If this is set (included in the file) `tfm` is forced to use full updates of the distance matrix instead of an approximate neighbour list. Unset by default. |
| **Protocol-specific options** | |
| `RMS_COEFF` *value* | `quick` protocol<br>Threshold for accepting the result of a `quick` simulation. Expressed as the number of residues per 1 Å RMSD: for example, a value of 50 would mean that a 200-residue sequence would have a threshold of 4 Å RMSD. This refers to the superposition of the simulated structure on the input template, and simulations not meeting the threshold are assumed to be poorly converged and will be repeated. Default: 75. |
| `MAX_COUNT` *n* | `quick` protocol<br>Sets the number of attempted simulations that will be carried out before failure is reported. Default: 3. |
| `SAMPLING` *n* | `strand_pair` protocol<br>Sets the number of distinct strand-pairing combinations to be run, and therefore also the number of output structures. Possible values are 0 (low), 1 (medium), and 2 (high), corresponding to the option menu in Maestro. The numbers are: low: 5, medium: 20, and high: 100. Default: 1. |

Table 13.7. Keywords for the refine input file (Continued)

| Keyword syntax | Description |
|---|---|
| `MIN_STRANDS` *n* | `strand_pair` protocol<br>Minimum number of strands allowed in a sheet. Prevents the creation of new sheets with fewer than the specified number of strands. Default: 1, i.e. unpaired strands could be left over subject to the other constraints. |
| `MAX_STRANDS` *n* | `strand_pair` protocol<br>Maximum number of strands allowed in a sheet. Prevents the creation of new sheets with more than the specified number of strands. Default is the total number of strands. |
| `NUM_SHEETS` *n* | `strand_pair` protocol<br>Desired number of sheets. The protocol will attempt to generate the number of strand pairings that will result in the specified total number of sheets. Specifying a reasonable number requires knowing the number of existing sheets and unpaired strands in each domain. Default is as few as possible, to maximize the number of new strand pairs. |
| **Global scoring options** | |
| `ALIGN_FILE` *filename* | Alignment file corresponding to a composite template. |
| `FR_FILE` *filename* | Name of a file containing the fold recognition scores for each seed template. |
| `SSP_FILE` *filename* | Name of a file containing all of the secondary structure predictions used in the fold-recognition step (i.e. not the assigned secondary structure of a given composite template.) It is used to assign the secondary-structure class in the global scoring. |
| `SSTYPE` *class* | Explicit specification of secondary-structure class. The default is `auto`, in which case `SSP_FILE` will be used to determine the class automatically. This can be overridden by setting `SSTYPE` to `alpha`, `beta`, or `mixed`. |

Table 13.8. Refinement protocols

| Protocol | Description |
|---|---|
| `default` | Basic simulation only. |
| `select` | Includes three `default` simulations and global scoring. This protocol is designed to give the most reliable scores, and corresponds to the Minimize (accurate) option in Maestro. |

*Table 13.8. Refinement protocols*

| Protocol | Description |
|---|---|
| `rough` | Uses modified run-time parameters to run a shorter simulation than `default` but includes the global scoring step as in `select`. Corresponds to the Minimize (fast) option in Maestro. |
| `fast` | Similar to `rough`, but even more stripped-down. This is the protocol used to produce global scores in the Build Backbone step. |
| `quick` | Fastest possible minimization of the structure to an input template, with no further optimization. Used in the Build Backbone step to generate structures for preliminary screening but without generating a global score. |
| `force` | Similar to `quick`, but with modified run-time parameters to produce a better result in some cases. Unlike `quick`, which can fail to converge, `force` is guaranteed to return a structure. Called from the Build Backbone step if `quick` fails to converge. |
| `strand_pair` | Corresponds to the Beta-strand Pairing option in Maestro. Consists of an initial run of `tfm` to generate possible pairings of unpaired strands, using a combinatorical search subject to topology constraints and scored by hydrophobic contacts. It then carries out simulations equivalent to `default` for each selected sheet topology and returns one structure for each. Simulations of different sheet topologies can be carried out in parallel. |

## Files

Refine produces the following files, where *root* is the jobname or structure name (depending on how the input is specified).

### Produced for each input composite template

These files are produced by the pre-processing of the template and are the same for any protocol

| | |
|---|---|
| *root*`.hbs` | Output from the program that parses the H-bond list. |
| *root*`.spair` | The actual strand-pair assignments as read in by `tfm`. |

### Produced for each run

These file names can include a “run code” indicating the protocol used, with a unique serial number.

| | |
|---|---|
| *root*.log | Output from the refine driver script. |
| *root*.err | Error messages only. |
| *root*.tab | Table of properties for each structure, as shown in the GUI. Each output structure is distinct from its input template and will have a protocol code appended to its entry name. The properties in the table are described in Table 13.9. Note: the name of this file can be overridden with the input option RESULT_TABLE. |
| *root*.out | In the case of protocols producing multiple structures, this file (if present) is the output from a simulation step refering to the whole set. Otherwise, see below. |

*Table 13.9. Properties generated for each structure*

| Property | Description |
|---|---|
| Rank score | The result of the global scoring function, the number that best represents the overall quality of the structure. This number should be used to rank the results. It is designed to be transferable between templates, and therefore allows you to select the best predictions among all simulations run for a given target. The actual simulation energies are dependent on secondary structure assignments, domain boundaries, etc, and cannot reliably be compared between different alignments. |
| Potential energy Z-score | The statistical Z-score of the target sequence compared to randomly shuffled sequences of the same composition evaluated for the same structure. It is a measure of the sequence-structure compatibility. |
| Sheet score | The score associated with the H-bond and local template constraints derived from the strand pairing. For the regular minimization protocols it is normalized in such a way that the maximum value is 2, but for the strand-pairing protocol it is a raw number that reflects the number (as well as quality) of the constraints. |
| Template score | The score associated with the superposition of the final structure on the original composite template. This score reflects the quality of the alignment, but also the relaxation of the structure away from the template during energy minimization and therefore also the compatibility of the template coordinates and the other constraints. |

### Produced for each output structure

For protocols (such as `strand_pair`) that generate multiple structures these files exist for each structure with an additional serial number added to *root*, otherwise they will only appear once as in the previous section.

| | |
|---|---|
| *root*`.pdb` | Contains the new predicted structure, and can be imported into Maestro as a regular structure. Contains the full target sequence (at least as used by the previous step) but backbone (+CB) atoms only. Contains SHEET/HELIX records that correspond to the input secondary structure assignment and input strand-pairing data. This information is not recalculated based on the actual coordinates produced and is independent of whether or not the constraints were actually satisfied. |
| *root*`.out` | Output from `tfm` itself for the simulation that generated the corresponding structure. |
| *root*`.sheethp` | Output from the global scoring program that evaluates hydrophobic contacts in beta sheets. Not present for all-alpha targets. |
| *root*`.spair` | In cases where the strand-pairing assignments are modified from that of the input template (see above), these are the input files corresponding to each output structure. |

### Return Value and Errors

`refine` returns a value of 0 on successful completion, 1 otherwise. All scripts run by `refine` that return a value of 1 also print a message to the file *jobname*`.err`. This file is empty otherwise, as all non-fatal messages are printed to the file *jobname*`.log`. Output from Job Control goes to standard output, and any errors arising before the driver script has started likewise go to standard output or standard error.

### Environment Variables

In addition to the standard Schrödinger environment variables `SCHRODINGER` and `SCHRODINGER_TMPDIR`, `refine` recognizes the following environment variables:

| | |
|---|---|
| `TMPDIR` | Directory where temporary pipes are created for inter-process communication. Default is /tmp. |

| | |
|---|---|
| PSP_RB_DEBUG<br>PSP_DEBUG | If this environment variable is set, debugging will be turned on and the subdirectory containing intermediate files will be kept. Equivalent to using the command-line option -DEBUG. |
| PSP_RB_LOCAL<br>PSP_LOCAL | If this is set, all calculations will run in the local launch directory. Equivalent to using the command-line option -LOCAL. |
| TFM_QUIET<br>TFM_VERBOSE | These environment variables can be set to decrease or increase the job-related output in the log file. |

### Examples

The following is an example of an input file produced by Maestro for a Minimize (Fast) calculation, called `prime_refine_bb_run1_tproj.inp`:

```
PROTOCOL            rough
USE_HB_SS           0
TEMP_DOMAIN         0
TEMPLATE_FILE_0     d1ewqb1_9.ent
TEMPLATE_NAME_0     d1ewqb1_9
FR_FILE             prime_refine_bb_run1_tproj-foldrec.out
SSP_FILE            prime_refine_bb_run1_tproj-ssp.seq
ALIGN_FILE          prime_refine_bb_run1_tproj-ali_d1ewqb1.9
```

This corresponds to composite template #9 with seed template `d1ewqb1`. The final three files are copied from earlier steps in the Prime Threading workflow: `foldrec.out` and `ssp.seq` from theFold Recognition step and `ali_d1ewqb1.9` from the `d1ewqb1` subdirectory of the Build Backbone step. The template file `d1ewqb1_9.ent` is a PDB format file copied from the Build Backbone step where it is called `temp_d1ewqb1_9.pdb`.

## 13.7 refinestruct—Refine a Protein Structure

The `refinestruct` script runs protein structure refinement jobs (side chain prediction and energy minimization as well as loop and helix prediction) using the Protein Local Optimization Program (PLOP).

`refinestruct` handles all the tasks associated with the refinement of protein structures via PLOP: parsing and preparation of input files, interaction with PLOP, and handling of output structures, etc. All jobs handled by `refinestruct` are run under Schrödinger Job Control.

### Syntax

```
$SCHRODINGER/refinestruct jobname
```

The input is read from the file *jobname*`.inp` and the structural output is written to the file *jobname*`-out.mae`.

## Command Input File

The command input file contains a list of keywords that control the operation of the `refinestruct` script, one per line. The keywords are listed in the tables below. Generally valid keywords are listed in Table 13.10 and protocol-specific keywords are listed in Table 13.11, under the protocol to which they apply.

*Table 13.10. General options for the refinestruct script*

| Keyword syntax | Description |
|---|---|
| `QUERY_FILE` *filename* | Input structure file in Maestro format. |
| `PROTOCOL` *n* | Type of refinement to perform. The available options are:<br>0 Predict Side Chains (protocol prefix `SC`)<br>1 Predict Loops and Helices<br>2 Minimize Energy (protocol prefix `MINI`)<br>3 Active Site Minimization (including ligand)<br>4 Energy Calculation<br>5 Protonation State Assignment (protocol prefix `PROT`) |
| `NUM_OUTPUT_STRUCT` *n* | Number of conformations to return for single loop refinements. This has no effect on helix refinements or jobs composed of more than one refinement (e.g. 2 loops, 1 helix + 1 loop, etc.). All structures are returned in a single structure file in Maestro format. Default: 1. |
| `ECUTOFF` *value* | Energy cutoff for return of conformations. Returns all conformations within *value* kcal/mol of the lowest-energy structure. It can be used in conjunction with `NUM_OUTPUT_STRUCT`, to impose a maximum on the number of such conformations to return. As with `NUM_OUTPUT_STRUCT`, it only applies to single loop refinements. |
| `USE_CRYSTAL_SYMMETRY {true\|false}` | Set to true if the input structure contains unit cell information and crystal symmetric atoms are to be included in calculation. Default: `false`. |
| `USE_RANDOM_SEED {true\|false}` | Indicates whether or not to use a random seed for the random number generator. This keyword has no effect on Minimization tasks. |
| `SEED` *n* | The integer to use as the seed for the random number generator if `USE_RANDOM_SEED` is false. If it is true, `SEED` is ignored. Default: -1, meaning that the program will generate a random seed rather than use the supplied seed. |

*Table 13.11. Protocol-specific keywords for the refinestruct script*

| Keyword syntax | Description |
|---|---|
| *prefix*_PROTO *string* | Specify the manner for specifying which residues to refine. Available options are:<br>ALL Refine all residues<br>PICK Refine the specific residues, indicated by included *prefix*_RESIDUE_*i* parameters (see below).<br>FILE Read list of residues to refine from a file. The file consists of a series of lines with a residue specification on each line. |
| *prefix*_RESIDUE_*i* *spec* | Specify a residue to refine. The numbering *i* begins at 0, and increases incrementally. The format of the residue specification *spec* is given above. |
| **PROTOCOL 0** | |
| NUM_SC_OUTPUT_STRUCT *n* | Same as NUM_OUTPUT_STRUCT above, as applied to sidechain prediction. Default 1. |
| **PROTOCOL 1** | |
| LOOP_*i*_RES_*j* *spec* | Specify residue *j* for defining loop *i*. Loops are numbered sequentially, beginning at 0. Two occurrences of this keyword are required, with *j*=0 (the beginning of the loop) and *j*=1 (the end of the loop). *spec* is a residue specification as defined above. |
| HELIX_*i*_RES_*j* *spec* | Specify residue *j* for defining helix *i*. Helices are numbered sequentially, beginning at 0, as are the residues themselves. Four residues are required in total; 1 and 2 define the limits of the helix itself, while 0 and 3 define the N-terminal and C-terminal loops, respectively. *spec* is a residue specification as defined above. |
| MAX_CA_MOVEMENT *value* | Specify the maximum distance that any alpha carbon atom in the loop should be allowed to move during refinement. Omitting this parameter indicates that no restriction should be applied. |
| RES_SPHERE *value* | All side chains that lie within this distance of the loop will also be optimized during refinement. This allows these nearby side chains to “react” to the loop being predicted. Default 0.0. |
| MIN_OVERLAP *value* | Fraction of van der Waals distance used to define a clash. Smaller values allow structures with worse potential atomic overlaps to be generated. Default: 0.70 |
| HELIX_*i*_ROLL_RANGE *value* | The maximum degree of sampling in the “roll” degree of freedom for helix *i*. |
| HELIX_*i*_ROLL_RES *value* | The increments (resolution) in which to sample the “roll” degree of freedom of helix *i*. |

*Table 13.11. Protocol-specific keywords for the refinestruct script (Continued)*

| Keyword syntax | Description |
|---|---|
| `HELIX_`*i*`_TILT_RANGE` *value* | The maximum degree of sampling in the “tilt” degree of freedom of helix *i*. |
| `HELIX_`*i*`_TILT_RES` *value* | The increments in which to sample the “tilt” degree of freedom of helix *i*. |
| `HELIX_`*i*`_SHIFT_RANGE` *value* | The maximum degree of sampling in the “shift” degree of freedom of helix *i*. |
| `HELIX_`*i*`_SHIFT_RES` *value* | The increments in which to sample the “shift” degree of freedom of helix *i*. |
| `HELIX_`*i*`_TRANS_RANGE` *value* | The maximum degree of sampling in the “translation” degree of freedom of helix *i*. |
| `HELIX_`*i*`_TRANS_RES` *value* | The increments in which to sample the translation of helix *i*. |
| **PROTOCOL 3** | |
| `LIGAND` *string* | Residue specification for the ligand. |
| `NPASSES` *n* | Number of passes through side chain optimization of the protein residues followed by minimization of the protein residues (including the backbone). These two steps are repeated the specified number of times before proceeding to optimization of the protein and the ligand. Default: 2 |
| **PROTOCOL 5** | |
| `RESIDUE_`*i* *spec* | Specifies residue to include in assignment. Same as *prefix*`_RESIDUE_`*i* defined above. |

## Files

In addition to the input structure file, defined by the value of `QUERY_FILE`, and the job files (the command input file *jobname*`.inp`, and the log file *jobname*`.log`), `bldstruct` produces the following output files:

| | |
|---|---|
| *jobname*`-out.mae` | Final structures, all in a single file. |
| *jobname*`.psane` | ProSane diagnostic output for the lowest-energy structure. Other structures have corresponding files *jobname*`-2.psane`, and so on. |

## Return Value

refinestruct returns a value of 0 on successful completion, 1 otherwise. All error messages are printed to the file *jobname*.log. There is no comprehensive listing of possible errors as these can arise in a variety of programs and scripts.

## Environment Variables

In addition to those used for all Schrödinger software, refinestruct uses the environment variable PSP_OPLS_VERSION to define the OPLS force field version. The default value is 2001. It can be set to 2005 to use the most recent force field parameters for ligands and non-standard residues.

## Examples

The following is a sample minimization input file:

```
QUERY_FILE        input-structure.mae
PROTOCOL          2
MINI_PROTO        PICK
MINI_RESIDUE_0    CHAINA:1
MINI_RESIDUE_1    CHAINA:20
MINI_RESIDUE_2    CHAINA:23
MINI_RESIDUE_3    CHAINA:26
MINI_RESIDUE_4    CHAINA:27
MINI_RESIDUE_5    CHAINA:30
MINI_RESIDUE_6    CHAINA:31
MINI_RESIDUE_7    CHAINA:36
MINI_RESIDUE_8    CHAINA:40
MINI_RESIDUE_9    CHAINA:42
MINI_RESIDUE_10   CHAINA:53
USE_RANDOM_SEED   false
SEED              0
```

The following is a sample loop/helix refinement input file:

```
QUERY_FILE              input-structure.mae
PROTOCOL                1
HELIX_0_RES_0           CHAIN_:101
HELIX_0_RES_1           CHAIN_:103
HELIX_0_RES_2           CHAIN_:112
HELIX_0_RES_3           CHAIN_:114
HELIX_0_ROLL_RANGE      120.00
HELIX_0_ROLL_RES        60.00
LOOP_0_RES_0            CHAIN_:40
LOOP_0_RES_1            CHAIN_:43
LOOP_1_RES_0            CHAIN_:45
LOOP_1_RES_1            CHAIN_:47
MAX_CA_MOVEMENT         5.00
RES_SPHERE              2.00
FIX_SIDE_CHAINS         false
NUM_OUTPUT_STRUCT       2
ECUTOFF                 10.00
USE_RANDOM_SEED         false
SEED                    0
```

### Restrictions

Helix refinement and protonation state assignment are not fully supported in this release.

## 13.8 ska—Protein Structure Alignment

The SKA program carries out structural alignment of proteins by aligning protein backbones at different levels of detail.

The `ska` script provides an interface to the SKA structural alignment program. As with other computational programs in Prime its operation is controlled through simple input files. The allowed keywords and their values are discussed in more detail below. The `ska` script allows a user to run the SKA program in three different modes under Schrödinger Job Control:

- **align mode**—SKA aligns two or more protein structures to one another
- **dbcreate mode**—SKA creates databases for use with scan mode
- **scan mode**—SKA searches a database of protein structures and identifies potential structural analogues. By default a non-redundant database of protein chains from the PDB will be searched.

## Syntax

```
$SCHRODINGER/ska [options] jobname
```

By default the input is read from the file *jobname*`.inp` and the output is written to *jobname*`.out`.

`ska` does not generate output other than the explicitly defined database and a log file in `dbcreate` mode—see below for details.

## Options

The only option accepted by `ska` is `-DEBUG`, which turns on debugging mode for both Job Control and the SKA program.

## Command Input File

The input file consists of lines containing keyword-value pairs, one per line. The keywords are described in Table 13.12. The set of allowed and required keywords depends on the mode, which is determined by the `MODE` keyword.

*Table 13.12. Keywords syntax for ska input file*

| Keyword syntax | Mode | Description |
|---|---|---|
| `MODE` *mode* | all | The SKA operating mode; allowed values are:<br>`*align*`: structurally align two or more protein structures to on another. This is the default mode<br>`*scan*`: given a query structure scan a database of protein structures and identify potential structural homologues<br>`*dbcreate*`: create a database that can be search in scan mode from a list of PDB files |
| `QUERY_FILE` *filename* [*name*] | `*align*` `*scan*` | Required. The input query structure. In `*align*` mode, the structure against which other proteins are aligned. In `*scan*` mode, the structure used for scanning the database. The optional *name* (`*align*` mode only) sets the name of this structure to a value other than the filename (the default). The name is displayed in the output file (see Examples, below). *name* cannot contain the file separator character ('/' on UNIX). |

*Table 13.12. Keywords syntax for ska input file (Continued)*

| Keyword syntax | Mode | Description |
|---|---|---|
| `TEMPLATE` *filename* [*name*] | `*align*` | Required. PDB structures that are to be aligned to the query structure, set by `QUERY_FILE`. The `TEMPLATE` keyword can appear multiple times in order to be able to carry out multiple structure alignments (aligning more than one template structure to the query structure). The optional *name* sets the name of this entry to a value other than the filename (the default). The name is displayed in the output file (see Examples, below). *name* cannot contain the file separator character ('/' on UNIX). |
| `DATABASE` *filename* | `*scan*` | The absolute path to the database to scan with the query structure. The database can be generated in dbcreate mode. Default: database supplied with Prime. |
| `DBNAME` *filename* | `*dbcreate*` | The name of the file in which to store the database create by `ska`. *filename* must be the file name without any path specification. |
| `FILELIST` *filename* | `*dbcreate*` | A file containing a list of absolute filenames for structures (in PDB format) that are to be included in a database of templates that can be search with `ska` in scan mode. Absolute filenames are required to avoid having to copy large numbers of PDB files to the temporary directory in which the job is run. |
| `GAP_OPEN` *value* | `*align*` `*scan*` | Gap initiation penalty. |
| `GAP_DEL` *value* | `*align*` | Deletion penalty. |
| `OUTPUT_FILE` *filename* | `*align*` `*scan*` | File in which to store the structural alignnment. Default is *jobname*`.out`. |
| `ALISTRUCS_OUTFILE` *filename* | `*align*` | File in which to store the 3D coordinates of the aligned structures. |
| `PSD_THRESHOLD` *value* | `*align*` `*scan*` | PSD threshold for inclusion in the MSA. Overriden by `RECKLESS`. |
| `SSE_MINSIM` *value* | `*align*` `*scan*` | Minimum Secondary Structural Element (SSE) similarity for the alignment. |
| `SSE_MINLEN` *value* | `*align*` `*scan*` | Minimum SSE length for the alignment. |
| `SSE_WINLEN` *value* | `*align*` `*scan*` | Align only SSE windows of length *value*. |

*Table 13.12. Keywords syntax for ska input file (Continued)*

| Keyword syntax | Mode | Description |
|---|---|---|
| `SWITCH [true|false]` | `*align*` `*scan*` | Carry out global alignment first and then align subwindows if no similarity was found. |
| `ORDER [seed|psd]` | `*align*` | `psd`: order the alignment by PSD to the seed structure. |
| `CLEAN` | `*align*` | Exclude structures with a `PSD` < `PSD_THRESHOLD` from the alignment. |
| `RECKLESS [true|false]` | `*align*` `*scan*` | Force the alignment of structures even if no similarity is found. |

## Return Value and Errors

The script returns 0 for success and 1 for failure.

In case of failure, check the log file for details on what might be have caused the program to fail. To turn on debugging output use the `-DEBUG` command line option as follows:

```
$SCHRODINGER/ska -DEBUG jobname
```

## Environment Variables

The `ska` program handles all required environment variables internally and requires only that the environment variable `SCHRODINGER` be set. Should the environment variable `TROLLTOP` exist in the current environment it will be overwritten.

## Examples

The following minimal input file (assumed to be called `align.inp`) structurally aligns two protein structures to one another:

```
QUERY_FILE query.pdb
TEMPLATE template.pdb
```

where `query.pdb` is the name of the PDB file containing the query structure (the structure to which we want to align) and `template.pdb` contains the template structure (the structure we want to align to the query).

When run using `$SCHRODINGER/ska align` it should produce the following output:

- `align.out`—a file containing the structural alignment
- `align.log`—a log file that contains log information and diagnostics on the run

Note that the actual 3D coordinates of the superposed structures are not written out by default You must use `ALISTRUCS_OUTFILE` to write out the structures.

The following input file (`align.inp`):

```
QUERY_FILE query.pdb
TEMPLATE template.pdb
ALISTRUCS_OUTFILE superposition.pdb
```

saves the 3D coordinates of the aligned structures in the file `superposition.pdb`.

The following is a sample input file for database creation in dbcreate mode:

```
MODE dbcreate
FILELIST filelist.dat
DBNAME mydatabase.dat
```

The file `filelist.dat` contains the file names for the templates, including the path, as follows:

```
/scr/templates/template-1.pdb
/scr/templates/template-2.pdb
/scr/templates/template-3.pdb
/scr/templates/template-4.pdb
```

A minimal input file for a scan is as follows:

```
MODE scan
QUERY_FILE query.pdb
```

Running ska with this input file would scan the database supplied with Prime for structural analogs. The output from the structural search contains pairwise alignments between the query structure and every structure in the database (assuming the two structures are sufficiently similar). The results can be analyzed in a more convenient way with the graphical SKA Results Viewer utility, `$SCHRODINGER/utilities/SkaResultsViewer`.

The use of the optional *name* argument is illustrated next. For the following input file:

```
QUERY_FILE query.pdb 1ctf
TEMPLATE 1dd3_A.pdb
```

the output file is as follows:

```
             .........+.........+.........+.........+.........+.........+.....
 1ctf    1                                                          ceeeeeeecc
1dd3_A    1  cchhhhhhhhhhchhhhhhhhhhhhhhhhccchhhhhhhhhhhhhhhhhhhhhccceeeeeeecc
 1ctf    1  ---------------------------------------------------------EFDVILKAA
1dd3_A    1  MTIDEIIEAIEKLTVSELAELVKKLEDKFGVTAAAPVAVAAAPVAGAAAGAAQEEKTEFDVVLKSF
```

```
            ..........+.........+.........+.........+.........+.........+.
 1ctf   62  ccchhhhhhhhhhhhcchhhhhhhhhhhc c  eeeeecchhhhhhhhhhhhhhcceeeeee
1dd3_A   67  ccchhhhhhhhhhhhcchhhhhhhhhhhcccccccceecchhhhhhhhhhhhhhcceeeeee
 1ctf   62  GANKVAVIKAVRGATGLGLKEAKDLVESA-P--AALKEGVSKDDAEALKKALEEAGAEVEVK
1dd3_A   67  GQNKIQVIKVVREITGLGLKEAKDLVEKAGSPDAVIKSGVSKEEAEEIKKKLEEAGAEVELK


RMSD: 0.763733
PSD: 0.024524
```

Without the *name* argument for the `QUERY_FILE` keyword, the output would be

```
.........+.........+.........+.........+.........+.........+.....
query    1                                                           ceeeeeecc
1dd3_A    1  cchhhhhhhhhhchhhhhhhhhhhhhhhhhcccchhhhhhhhhhhhhhhhhhhhhcccceeeeeeecc
query    1  ---------------------------------------------------------EFDVILKAA
1dd3_A    1  MTIDEIIEAIEKLTVSELAELVKKLEDKFGVTAAAPVAVAAAPVAGAAAGAAQEEKTEFDVVLKSF

            ..........+.........+.........+.........+.........+.........+.
query   62  ccchhhhhhhhhhhhcchhhhhhhhhhhc c  eeeeecchhhhhhhhhhhhhhcceeeeee
1dd3_A   67  ccchhhhhhhhhhhhcchhhhhhhhhhhcccccccceecchhhhhhhhhhhhhhcceeeeee
query   62  GANKVAVIKAVRGATGLGLKEAKDLVESA-P--AALKEGVSKDDAEALKKALEEAGAEVEVK
1dd3_A   67  GQNKIQVIKVVREITGLGLKEAKDLVEKAGSPDAVIKSGVSKEEAEEIKKKLEEAGAEVELK


RMSD: 0.763733
PSD: 0.024524
```

## 13.9 ssp—Secondary Structure Prediction

The `ssp` script provides an interface to the secondary structure prediction programs SSPro, which is bundled with Prime, as well as PSIpred and Prof, which are available from third parties.

### Syntax

`$SCHRODINGER/ssp [`*options*`]` *jobname*

By default the input is read from the file *jobname*`.inp` and the output is written to *jobname*`.out`.

### Options

The standard Job Control command line options listed in Table 13.1 are accepted. There are no options specific to this program.

## Command Input File

The command input file consists of lines containing keyword-value pairs, one per line. The keywords are described in Table 13.13.

*Table 13.13. Keywords for the ssp input file.*

| Keyword syntax | Description |
|---|---|
| `QUERY_FILE` *filename* | Required. Name of the FASTA file containing the query sequence |
| `METHOD` *program* | Required. Name of the secondary structure prediction that is to be run. Allowed values are `sspro`, `psipred`, `profking`, and `all`. `all` means run all available programs. If the input file contains multiple `METHOD` keywords, only the last one is used. |
| `OUTPUT_FILE` *filename* | Optional. Name of the output file. Default is *jobname*`.out`. |

## Return Value and Errors

`ssp` returns 0 if successful and a non-zero value in all other cases.

## Environment Variables

The environment variables that can be used to control the secondary structure prediction programs are listed in Table 13.14.

*Table 13.14. Environment variables for secondary structure prediction programs.*

| Environment Variable | Description |
|---|---|
| `PSP_SSPRO_DB` | BLAST database to be searched when assembling the multiple sequence alignment used in the SSPro prediction |
| `PSP_PSIPRED_DB` | BLAST database to be searched when assembling the sequence profile used in the PSIpred prediction |
| `PSP_PROFKING_DB` | BLAST database to be searched when assembling the sequence profile used in the Prof (King) prediction |
| `PSP_PROFKING_NSEQ` | The number of sequences to include in the multiple sequence alignment (MSA) use in the Prof (King) prediction. Note that including large numbers of sequences in the MSA, which is generated by ClustalW, can lead to performance degradation. |

### Examples

The following minimal input file, called `sspro.inp`, generates an SSPro secondary structure prediction for the sequence stored in file `query.fasta`:

```
QUERY_FILE query.fasta
METHOD     sspro
```

The prediction produced by the back-end is written to the file `sspro.out`.

The following input file, assumed to be called `all-methods.inp`, instructs the `ssp` script to produce secondary structure prediction for all methods available—SSPro and, depending on the local installation, PSIpred and Prof (King).

```
QUERY_FILE query.fasta
METHOD all
```

## 13.10 sta—Single Template Alignment

The STA program in PRIME is designed for protein sequences with medium to high sequence identity (>25%). "Single Template Alignment" was initially used to distinguish itself from "composite template alignment" used in Build Backbone. STA differs from standard sequence alignment such as BLAST by taking into account secondary structure matching as well as profile-sequence mathcing. As a result, PRIME-STA often generates better alignment than BLAST in the regions where sequence conservation is relatively weak. STA can also take in constraints, such as user-selected residue pairs to be aligned together and/or residues in either query or template that are not to be aligned (gaps).

STA can use templates directly from the PDB, as selected by running the Find Homologs step, and can also use templates from the Prime fold library, as selected in the Fold Recognition step. For the latter, you can export the template from the Fold Recognition table, and save it as a PDB file.

### Syntax

`$SCHRODINGER/sta [`*options*`]` *jobname*

The input is read from the file *jobname*`.inp` and the output is written to *jobname*`.out`.

### Options

The `sta` script accepts the `-DEBUG` and `-HOST` options, as described in Table 13.1.

## Command Input File

The input file consists of lines containing keyword-value pairs, one per line. The keywords are described in Table 13.15.

*Table 13.15. Keywords for the sta program.*

| Keyword syntax | Description |
|---|---|
| `QUERY_NAME` *name* | Name of the query sequence. |
| `QUERY_FILE` *filename* | File name of the query sequence in FASTA format. |
| `FORMAT` *string* | Alignment file format. Default is Maestro. Set to `dev` for plain text format—see page 131 for an example. |
| `BGNRES` *n* | The first residue of the query that is used in alignment. Default: 1. |
| `ENDRES` *n* | The last residue of the query sequence that is used. Default: the last residue of the original query sequence. |
| `PREDSS[_`*i*`]` *filename* | Secondary structure prediction file of the query sequence, in CASP format. One occurrence of this keyword is required for each file, and there must be at least one SSP file specified. If only one prediction is used, the keyword is `PREDSS`; if more predictions are used, the keyword is suffixed with an index *i*, starting from zero: `PREDSS_0, PREDSS_1`, and so on. For an example of CASP format, see page 134. |
| `TEMPLATE_PDB` *filename* | Filename of the template structure, in PDB format. |
| `MODE` *string* | Optional. Needed in the input file only when there are user-specified constraints. In that case, the value is set to `user`. |
| `PAIR` *pair-list* | Optional. User-specified residue pairing, used only when `MODE` is set to `user`. All the residue pairs can be added after the keyword in one or multiple lines. For example, the following lines pair residue 12, 20, 60 from the query sequence with residue 15, 24, 65 from the template.<br>`MODE user`<br>`PAIR 12 15 20 24`<br>`PAIR 60 65` |
| `GAP` *gap-list* | Optional. User-specified gaps, used only when `MODE` is set to `user`. Can be used with or without `PAIR`. Gaps are specified by the residues in either query (P) or template (T) that are aligned with them. For example, the following opens gaps against query residue 10, 11, template residue 34 and 35.<br>`MODE user`<br>`GAP  P10 P11 T34 T35` |

## Files

- Input file—named *jobname*`.inp`. Contains keywords for running the program.
- Query sequence file—File containing the complete sequence of the query, in FASTA format. Specified in the input file.
- Secondary structure prediction files in CASP format for the query sequence. See page 134 for an example of CASP format.
- Template PDB file—Template structure file, in PDB format. The sequence used for the template is taken from `SEQRES` lines in the PDB file. If `SEQRES` is not found in the PDB file, the template sequence is then taken from `ATOM` records.
- Output alignment file—File named *jobname*`.out` containing pairwise alignment between the query and the template. By default, the alignment is in Maestro format. To generate plain text format, include `FORMAT dev` in the command input file. In plain text fomat, aligned residues from the query and the template are placed on top of each other—see page 131 for an example.
- Log file—File named *jobname*`.log`containing progress of the `sta` program, including warnings and error messages.

## Return Value and Errors

`sta` returns 0 for success and non-zero for failure. In case of failure, check the log file for details, and search for `ERROR`. To further analyze the failure, use the `-DEBUG` option when you run `sta`.

## Examples

Below is an example input file, where `query_seq.fasta` stores the query sequence in FASTA format.

```
QUERY_NAME      query
QUERY_FILE      query_seq.fasta
PREDSS_0        query-ssp1.casp
PREDSS_1        query-ssp2.casp
PREDSS_3        query-ssp3.casp
BGNRES          1
ENDRES          149
TEMPLATE_PDB    pdb1vpe.ent
```

## 13.11 tfm—Tertiary Folding Module

The tfm program generates one or more complete backbone conformations starting from a primary sequence and secondary structure, along with optional domain boundaries and sheet topologies. In addition, `tfm` can accept several different types of constraint including multiple coordinate templates, multiple types of distance constraints, and dihedral angle constraints. A large number of run-time parameters allow the code to be used in a variety of ways, from minimizing a homology model to full ab initio folding.

### Syntax

The `tfm` program has hundreds of user-specified options contained in over a dozen input files. A detailed description of all the options is beyond the scope of this document. The `refine` command can be used as a wrapper to prepare default input files. If suitable input has been prepared, there is also a top-level wrapper for `tfm` that can be used to run the program directly under Schrodinger Job Control. It can be invoked from the command line via:

`$SCHRODINGER/tfm` *input-file* `[`*options*`]`

where *input-file* is typically the job name, with or without a `.inp` extension. The input file must be either the first or last command-line argument, and everything else on the command line is passed as is to Job Control.

### Options

All options are specified in the input file, and there are no command-line options, other than those recognized by Job Control.

### Command Input File

There are only three keywords recognized in the command input file:

*Table 13.16. Keywords for tfm program.*

| Keyword | Description |
|---|---|
| `INPUT_FILE` *filename* | The input file passed to the executable. |
| `OUTPUT_FILE` *filename* | The output produced by the executable. |
| `STRUCT_NAME` *name* | Optional. Prefix for files in the local directory that will be automatically treated as input, such as template or constraint files. |

## Files

In addition to the descriptive output in the file specified by `OUTPUT_FILE` and the error messages in *name*`.err`, `tfm` produces the following files, where the first part of the file name is `tfm` by default but can be changed by other input options, and *N* is a serial number.

| | |
|---|---|
| `tfm.`*N*`.pdb` | Output structures, if applicable. |
| `tfm.`*N*`.spair` | Possible strand pair assignments for unpaired strands, if automatic strand-pairing is being used. |

## Return Value and Errors

`tfm` returns a value of 0 on successful completion, 1 otherwise. All error messages are printed to the file *name*`.err`, where *name* is either the `STRUCT_NAME` value or the `SEQNAME` parameter in the file specified by `INPUT_FILE`. This file is empty otherwise, and all output from the program itself is written to the file specified by `OUTPUT_FILE`. Output from Job Control goes to stdout, and any errors arising before the driver script has started will likewise go to stdout or stderr.

## Examples

Sample input for `tfm` can be generated by running `refine` with the `-LOCAL` and `-DEBUG` options. The command

```
refine sample
```

creates a directory `run_`*sample* with the input file *sample*`.1.in` (among other files determined by the refinement protocol). An input file *sample*`_tfm.inp` containing the following lines can be used to run the job from the command line

```
INPUT_FILE sample.1.in
OUTPUT_FILE sample.1.out
```

with the command

```
tfm sample_tfm
```

## Environment Variables

In addition to the standard environment variables `tfm` recognizes the following environment variables:

| | |
|---|---|
| `TFM_QUIET`<br>`TFM_VERBOSE` | These environment variables can be set to decrease or increase the job-related output in the log file. The value to which they are set is irrelevant. |

# Command-Line Utilities

## 14.1 PDB Structure/Chain Extraction: `getpdb`

The `getpdb` utility extracts entire proteins or single chains from the PDB database. Extraction of single chains is useful when structural alignment of multiple templates encounters difficulties due to multiple-chain template files. See Section 4.2.5.

**Syntax:**

`getpdb` *PDB_code*`[:`*Chain_ID*`]`

**Note:** To use `getpdb`, you must have one of the following environment variables set:

- `$SCHRODINGER_PDB`

  which should point to the directory that contains `data/structures/divided/pdb/*/pdb*.ent.Z`.

- `$SCHRODINGER_THIRDPARTY`
- `$SCHRODINGER`

  in which case the default directory for the third-party database is used, `$SCHRODINGER/thirdparty`.

**To extract the entire protein from the PDB database, enter:**

`$SCHRODINGER/utilities/getpdb` *PDB_code*

For example:

```
$SCHRODINGER/utilities/getpdb 1aaa
```

retrieves the PDB file for the structure with PDB code 1AAA.

**To extract a single chain, enter:**

`$SCHRODINGER/utilities/getpdb` *PDB_code*`:`*Chain_ID*

For example:

```
$SCHRODINGER/utilities/getpdb 1ems:A
```

extracts all residues (including HETATMs) in chain A and HETATMs with no chain name.

**Note:** The *Chain_ID* variable is case sensitive, though the *PDB_code* variable is not. In this example, `1EmS:A` would also work, but `1ems:a` would produce errors.

## 14.2 Multiple Template Alignment: `structalign`

If multiple templates are selected in the Find Homologs step, they must be rotated into the same coordinate reference frame before they can be used. The Align Structures button is the usual means of performing structural alignment of templates, but the `structalign` utility can be run from the command line.

The input files must be in PDB format and the templates in them should consist of single chains. To use the output files, each must be imported into Find Homologs and selected as one of the templates, replacing the original, unaligned version.

### Syntax:

```
$SCHRODINGER/utilities/structalign [options] input-files
```

The input files must be in PDB format. The output file name is generated by adding the prefix `rot-` to the input file name.

### Options:

| | |
|---|---|
| `-h` | Print this usage message. |
| `-force` | Force alignment even if structures are not sufficiently similar. |

**To use `structalign` from the command line:**

1. Obtain the PDB structure file of each template. This can be done easily by clicking on the sequence name of each template of interest to display it in the Workspace.
2. For each structure, choose Export from the Project menu of the Maestro main window to save the coordinates of the structure to a file. Select PDB format.

   You should now have the following files:

   *template1*`.pdb` *template2*`.pdb` *template3*`.pdb`
3. At the command line, enter:

   ```
   $SCHRODINGER/utilities/structalign template1.pdb template2.pdb
   template3.pdb
   ```

   This command runs the `structalign` utility and returns one aligned file for each template:

```
rot-template1.pdb
rot-template2.pdb
rot-template3.pdb
```

**Note:** If you encounter a problem involving templates with multiple chains, use the `getpdb` utility (see Section 14.1) to extract each chain into a separate `.pdb` file, then run `structalign` again.

4. Import each `rot-`*template*`.pdb` file using the Import button.

Once the files are imported and the aligned templates appear in the Homologs table, select them all using the SHIFT-click or CTRL-click shortcuts. (Do not select the original structures.) Click Next to continue to the Edit Alignment step.

## 14.3 Parameter Generation: `hetgrp_ffgen`

**Syntax:**

```
$SCHRODINGER/utilities/hetgrp_ffgen version infile [outfile]
```

| | |
|---|---|
| *version* | OPLS force field version. Allowed values are `2001` and `2005` |
| *infile* | Input file (Maestro format) containing the structure |
| *outfile* | Output structure file (Maestro format) containing extra data |

**To generate missing parameters for a ligand or cofactor:**

1. Click Hide to undisplay the Prime–SP panel.
2. Clear the Workspace.
3. Import the ligand or cofactor that has missing parameters into the Workspace.
4. Open the Maestro Build panel by clicking the Show/Hide build panel toolbar button

   or selecting Build from the Maestro Edit menu.

5. Find and correct any bond orders that need to be changed, such as missing double bonds. You can set bond orders using the Increment bond order and Decrement bond order buttons in the Build panel toolbar.

6. Select Hydrogen Treatment from the Edit menu and apply the All-atom with No-Lp treatment to the ligand/cofactor.

7. Check the ligand or cofactor molecule.

   For parameter generation, it must have correct bond orders and hydrogens, and atom names must satisfy these conditions:

   - All atoms in the residue must have the same PDB Residue Name, Residue Number, and Chain Name. (Check this by creating labels with that composition and applying them to all atoms in the residue.)

     If they do not, then (for example):

     a. Click the Residue Properties tab in the Build panel.
     b. Select Residue Name from the Property list.
     c. Specify a Residue Name in the text box.
     d. Use the picking controls to Apply residue PDB name to the ligand or cofactor.

   - All atoms must have PDB Atom Names that are unique within the residue, as required for parameter generation.

     If they do not, click the Atom Properties tab in the Build panel, select PDB Atom Name from the Property list, and under Set unique PDB atom names within residues, select Pick and pick the ligand or cofactor residue.

8. Export the structure as a Maestro-format file (*filename*`.mae`) from the Workspace.

9. In the current directory, from the command line, enter:

   `$SCHRODINGER/utilities/hetgrp_ffgen` *filename*`.mae`

10. Open the Prime–SP panel and click Build to run the structure-building job again.

## 14.4 Format Conversion: `seqconvert`

### Syntax:

```
$SCHRODINGER/utilities/seqconvert [options]
input-format  input_filename  output-format  output_filename
```

One use of the `seqconvert` utility is to convert SSPs produced in Prime to FASTA format with the following command:

```
$SCHRODINGER/utilities/seqconvert -inative infile.m2io
-ofasta outfile.fasta
```

The `seqconvert` utility performs the conversion at the command line between any permitted input file format and output file format. This may be useful if autoconversion does not proceed as expected.

The command:

```
$SCHRODINGER/utilities/seqconvert -h
```

displays the usage message and the list of options, including accepted input and output file formats, shown in Table 14.1:

*Table 14.1. Keywords for the* `seqconvert` *utility*

| Option | Description |
|---|---|
| `-start` | Read from this sequence in the input file |
| `-end` | Read to (and including) this sequence in the input file |
| `-total` | Read at most this number of sequences |
| `-ali` | Read an alignment in the input file |
| `-pdb_use_atoms` | Use ATOM (not SEQRES) records with PDB files |
| *Input File Formats:* | |
| `-iany` | Autodetect input file format |
| `-inative` | Input file in native (Schrödinger) format |
| `-ifasta` | Input file in FASTA format |
| `-ipdb` | Input file in PDB format |
| `-inbrf_pir` | Input file in NBRF-PIR format |
| `-itext_plain` | Input file in TEXT-PLAIN format |

*Table 14.1. Keywords for the* `seqconvert` *utility (Continued)*

| Option | Description |
|---|---|
| `-igcg` | Input file in GCG format |
| `-iswissprot` | Input file in SWISS-PROT format |
| `-igenbank` | Input file in GENBANK format |
| `-incbi` | Input file in NCBI format |
| `-iphylip` | Input file in PHYLIP format |
| `-inexus_paup` | Input file in NEXUS-PAUP format |
| `-iselex` | Input file in SELEX format |
| `-istaden` | Input file in STADEN format |
| `-ipfam_stockholm` | Input file in PFAM-STOCKHOLM format |
| `-icasp` | Input file in CASP format |
| *Output File Formats:* | |
| `-onative` | Output file in native (Schrödinger) format |
| `-ofasta` | Output file in FASTA format |
| `-onbrf_pir` | Output file in NBRF-PIR format |
| `-otext_plain` | Output file in TEXT-PLAIN format |
| `-ogcg` | Output file in GCG format |
| `-oswissprot` | Output file in SWISS-PROT format |
| `-ocasp` | Output file in CASP format |
| `-ogenbank` | Output file in GENBANK format |

# Third-Party Programs

## 15.1 Location of Third-Party Programs

As well as internal and bundled software and data, Prime uses certain external third-party programs and databases for sequence and homolog searches, protein family searches, and secondary structure prediction. These are described under the headings in this topic.

### 15.1.1 BLAST/PSI-BLAST, Pfam, and the PDB

See the home pages for these programs and servers for more information:

- BLAST/PSI-BLAST

  http://www.ncbi.nlm.nih.gov/blast

  The BLAST databases distributed with Prime are formatted with `formatdb` version 2.2.6. You may experience problems with Prime if your database is not in this format.

- HMMER/Pfam

  http://pfam.wustl.edu

- PDB

  http://www.rcsb.org/pdb/

### 15.1.2 Secondary Structure Prediction

Two distinct third-party secondary structure prediction programs can be used to increase the accuracy of, and get better results from, Prime. One of these programs (SSpro) is already bundled with Prime. The other (PSIPRED) may be manually installed by you, at your election, and subject to applicable license terms, if any. At the third-party web site whose address is given below, you may download the program and then install it. Please see Section 15.3 or the Third Party Programs page of our website for instructions on how to compile and install third-party programs.

The PSIPRED home page is located at:

http://bioinf.cs.ucl.ac.uk/psipred/

and contains information about the program, referrals to terms of use and/or license terms (in the README file), and a link to download the program. The link to download PSIPRED is contained in the section of the page labeled “Overview of prediction methods.”

## 15.2 Third-Party Program Use Agreement

By using the links above or the third-party programs, you acknowledge and agree to the following:

PSIPRED is a third party program (“Third Party Program”) and may therefore be subject to third party license agreements and fees. We (i.e., Schrödinger, LLC and its affiliates) have no responsibility or liability, directly or indirectly, for Third Party Programs or for any third party Web sites (“Linked Sites”) or for any damage or loss alleged to be caused by or in connection with use of or reliance thereon. Any warranties that we make regarding our own products and services do not apply to the Third Party Programs or Linked Sites, or to the interaction between, or interoperability of, our products and services and the Third Party Programs. Referrals and links to Third Party Programs and Linked Sites do not constitute an endorsement of such Third Party Programs or Linked Sites.

## 15.3 Instructions for Compiling and Installing Third-Party Programs

Below you will find instructions for compiling and installing the third-party program PSIpred. It is assumed that the environment variable `SCHRODINGER` is defined and points to your installation of Prime, that you have already installed at least one of the Schrödinger third-party products (Blast, HMMER/Pfam, or PDB), and that the directory `$SCHRODINGER/thirdparty` exists.

### 15.3.1 Determining Your Architecture

The third-party program PSIPRED must be compiled for every platform (combination of operating system and processor architecture) on which you intend to use it. See Section 15.1.2 above for information on downloading third-party programs. Schrödinger software uses simple architecture tags to distinguish different versions.

You can determine your current platform with the following command:

```
$SCHRODINGER/platform -d
```

For example, the architecture tag for a Linux system running on Intel-architecture hardware is `Linux-x86`, the one for IRIX (SGI hardware) is `IRIX-mips4`.

In the instructions below, substitute your current platform for the generic *arch* tag.

## 15.3.2 Environment Variables

If you have relocated the `$SCHRODINGER/thirdparty` directory and are pointing to it with the `SCHRODINGER_THIRDPARTY` environment variable, substitute `$SCHRODINGER/thirdparty` with `$SCHRODINGER_THIRDPARTY` in the sections below.

## 15.3.3 Compiling and Installing PSIpred

After downloading the source code for PSIPRED (`psipred23.tar.gz`), follow these steps to make it available for use with Prime:

1. Unpack the source code in the build directory (below we refer to this directory as *build-dir*):

   ```
   gzip -d -c psipred23.tar.gz | tar xvf -
   ```

2. Change directory to `src`:

   ```
   cd src
   ```

3. Build the software by running `make`:

   ```
   make
   ```

4. Install the executables into the `bin` directory:

   ```
   make install
   ```

5. Create the PSIPRED install directory in the Schrödinger `thirdparty` directory structure (unless it already exists). Substitute your specific platform for *arch*:

   ```
   mkdir -p $SCHRODINGER/thirdparty/bin/arch/psipred
   ```

   For example, on a Linux machine with an Intel architecture processor you would have to create the following directory:

   ```
   mkdir -p $SCHRODINGER/thirdparty/bin/Linux-x86/psipred
   ```

6. Copy over the `bin` and `data` directories from the directory in which you compiled PSIpred:

   ```
   cp -R build-dir/bin $SCHRODINGER/thirdparty/bin/arch/psipred/
   cp -R build-dir/data $SCHRODINGER/thirdparty/bin/arch/psipred/
   ```

7. The final layout should look as follows:

   ```
   ls -R $SCHRODINGER/thirdparty/bin/arch/psipred/
   ```

```
.:

bin  data

./bin:

pfilt     psipass2     psipred     seq2mtx

./data:

weights.dat     weights.dat3     weights_p2.dat     weights_s.dat2

weights.dat2    weights.dat4     weights_s.dat      weights_s.dat3
```

# Getting Help

Schrödinger software is distributed with documentation in PDF format. If the documentation is not installed in `$SCHRODINGER/docs` on a computer that you have access to, you should install it or ask your system administrator to install it.

For help installing and setting up licenses for Schrödinger software and installing documentation, see the *Installation Guide*. For information on running jobs, see the *Job Control Guide*.

Maestro has automatic, context-sensitive help (Auto-Help and Balloon Help, or tooltips), and an online help system. To get help, follow the steps below.

- Check the Auto-Help text box, which is located at the foot of the main window. If help is available for the task you are performing, it is automatically displayed there. Auto-Help contains a single line of information. For more detailed information, use the online help.
- If you want information about a GUI element, such as a button or option, there may be Balloon Help for the item. Pause the cursor over the element. If the Balloon Help does not appear, check that Show Balloon Help is selected in the Help menu of the main window. If there is Balloon Help for the element, it appears within a few seconds.
- For information about a panel or the folder that is displayed in a panel, click the Help button in the panel. The Help panel is opened and a relevant help topic is displayed.
- For other information in the online help, open the Help panel and locate the topic by searching or by category. You can open the Help panel by choosing Help from the Help menu on the main menu bar or by pressing CTRL+H.

  To view a list of all available Prime–related help topics, choose Prime from the Categories menu of the Categories tab. Double-click a topic title to view the topic.

If you do not find the information you need in the Maestro help system, check the following sources:

- *Maestro User Manual*, for detailed information on using Maestro
- *Maestro Tutorial*, for a tutorial on the basic features of Maestro
- *Maestro Command Reference Manual*, for information on Maestro commands
- *Prime Quick Start Guide*, for a tutorial guide to using Prime
- Frequently Asked Questions pages, at https://www.schrodinger.com/Prime_FAQ.html

The manuals are also available in PDF format from the Schrödinger Support Center. Information on additions and corrections to the manuals is available from this web page.

If you have questions that are not answered from any of the above sources, contact Schrödinger using the information below.

| | |
|---|---|
| E-mail: | help@schrodinger.com |
| USPS: | 101 SW Main Street, Suite 1300, Portland, OR 97204 |
| Phone: | (503) 299-1150 |
| Fax: | (503) 299-4532 |
| WWW: | http://www.schrodinger.com |
| FTP: | ftp://ftp.schrodinger.com |

Generally, e-mail correspondence is best because you can send machine output, if necessary. When sending e-mail messages, please include the following information, most of which can be obtained by entering `$SCHRODINGER/machid` at a command prompt:

- All relevant user input and machine output
- Prime purchaser (company, research institution, or individual)
- Primary Prime user
- Computer platform type
- Operating system with version number
- Prime version number
- Maestro version number
- mmshare version number

# Environment Variables

The only environment variable that *must* be set before you can run Prime is `SCHRODINGER`. The following list is presented for your convenience in case you prefer a different location for any Prime-related directories.

## General

`SCHRODINGER_PDB`: Full path to a user-supplied copy of the PDB database. This copy must retain the directory hierarchy used by the PDB and must include the following sub-directories to be usable with Prime:

```
data/structures/all/pdb
data/structures/divided/pdb
data/structures/obsolete/pdb
```

`SCHRODINGER_THIRDPARTY`: Directory containing third-party software and databases (installed via the installers supplied by Schrödinger).

## Blast

`PSP_BLAST_DIR`: Location of Blast executables.

`PSP_BLAST_DATA`: Location of Blast matrices.

`PSP_BLASTDB`: Location of Blast databases. The BLAST databases distributed with Prime are formatted with `formatdb` version 2.2.6. If your BLAST database does not conform to this format, you might experience problems with Prime.

## HMMER/Pfam

`PSP_HMMER_DIR`: Location of HMMER distribution (the executables are assumed to be in the `binaries` subdirectory).

`PSP_HMMERDB`: Location of Pfam database.

## Psipred

`PSP_PSIPRED_DIR`: Location of Psipred installation (top level directory containing the `bin` and `data` directories).

`PSP_PSIPRED_DB`: Sequence database to use for assembling the PSI-BLAST profile.

## SSPRO

PSP_SSPRO_DB: Sequence database to use for assembling the PSI-BLAST profile.

# File Formats

## B.1 Input File Formats

The input file formats are:

- native (Schrödinger) format
- FASTA format
- PDB format

  Sequences are read from the PDB SEQRES records, if they exist. Otherwise they are read from the ATOM records. If SEQRES records do not exist and residues are missing from the ATOM records, the query sequence will of course contain gaps.

- NBRF-PIR format
- TEXT-PLAIN format
- GCG format
- SWISS-PROT format
- GENBANK format
- NCBI format
- PHYLIP format
- NEXUS-PAUP format
- SELEX format
- STADEN format
- PFAM-STOCKHOLM format
- CASP format

## B.2 Output File Formats

The output file formats are:

- native (Schrödinger) format
- FASTA format
- NBRF-PIR format
- TEXT-PLAIN format
- GCG format
- SWISS-PROT format
- CASP format
- GENBANK format

# Error Messages and Warnings

## C.1 Input Sequence

#### Invalid File

This message appears if the system fails to extract a valid sequence from the file.

- Check that the sequence is in one of the supported sequence file formats.
- If FASTA format is used, make sure there are no extra spaces between the ">" and the sequence name, and that there are no extra spaces at the end of the line.

#### No Sequences Found

This message appears if the system fails to extract any sequences from the workspace.

- Check the Project Table to ensure the correct entries are included in the Workspace.
- Use Open Project in the Maestro Window to open a project.

#### Invalid Sequence Characters

This message appears if the sequence contains non-standard one-letter code for amino acids.

- Check that sequence uses the one-letter code and is all uppercase.
- Change/edit any non-standard characters from your sequence using an editor, and then restart job.

## C.2 Find Homologs

### C.2.1 Error Messages

#### Multiple Templates Not Aligned

- If you select multiple templates and proceed to Edit Alignment, you are reminded that the templates must be structurally aligned to one another.

If one of the following messages appears when you click Next to go to the Edit Alignment step, you will not be able to proceed along the Comparative Modeling path until the problem is resolved:

#### No Homologs Found

- You may change your search options and continue searching, or proceed to Fold Recognition.

#### Invalid File

- Check to ensure the homolog file exists (in either `$SCHRODINGER/thirdparty/database/pdb/data/structures/divided/pdb/*/` or `$SCHRODINGER_THIRDPARTY/database/pdb/data/structures/divided/pdb/*/`) and is in the right format (UNIX compressed, `pdb*.ent.Z`).

### C.2.2 Warnings in Homologs Table

These messages appear in the Warning column of the Homologs table for sequences that are likely to be unusable for model building:

#### Many missing residues

- Too many consecutive missing residues to guarantee model building will succeed.

#### !UNUSABLE: Alternate residue type

- The residue type was undetermined, causing the file to include the residue twice but with different identity (e.g. ALA20 and VAL20). This is not supported by Prime.

#### !UNUSABLE: Bad SEQRES

- General problems with the SEQRES.

#### !UNUSABLE: CA only

- File contains only alpha C coordinates.

#### !UNUSABLE: Molecule *molname* should be labeled as HETATM

- A non-protein molecule has been listed as ATOM when it should be listed as HETATM. ATOM records are reserved for proteins only.

#### !UNUSABLE: SEQRES/ATOM mismatch

- The sequence listed in the SEQRES records does not match the actual sequence in the ATOM records.

#### !UNUSABLE: SEQRES is missing residues

- There are residues in the ATOM records that are not listed in the SEQRES records. PDB files require that the ATOM records be a subset of the SEQRES records.

**!UNUSABLE: SEQRES numbering is wrong**

- The number of residues in a particular chain does not agree with the number of residues shown in columns 14 to 17 of SEQRES.

## C.3 Edit Alignment

### Check Alignment

When you click Next at the end of the Edit Alignment step, the Check Alignment warning may appear. The warning lists possible alignment problems that you may want to fix before continuing to the next step. For example, there may be gaps in secondary structure elements of the template, or gaps in the query that are aligned to secondary structure elements of the template.

If you do not want to make corrections to the alignment, click Continue to go to the next step. To make corrections based on the information in the error message, make a note of the gap locations given, click Cancel to close the message box while remaining in Edit Alignment, and change the alignment accordingly.

## C.4 Build Structure

Error messages and warnings for this and several other steps (Refine Structure, Build Backbone, Refine Backbone, and the stand-alone Refinement panel) generally appear in the log of a running job. The log can be viewed in the job Monitor panel until the job is completed. The following is an example of a (rare) warning that may appear in the log of a Build Structure job.

### Missing Parameters

If you included a ligand or cofactor in your structure that is not in the provided database, it may be missing a force field parameter. When the Build job begins, it checks for missing parameters and reports them in its log. If you see this warning in the Build log, you may need to create new parameters for the ligand or cofactor. This can be done using the `hetgrp_ffgen` utility. To use this procedure, see Section 14.3.

## C.5 Refine Structure

Error messages for Refine Structure generally appear in the log of a running job. The following is an example of a warning in the log of a loop refinement.

## Invalid Features

When a loop refinement begins, a validation program checks the loop features of the structure. Apparent errors are reported in a warning dialog box. You may choose to Run All Features, Run Only Valid Features, or Cancel. It is recommended that you make a note of the invalid features, click Cancel, and correct the structure if appropriate.

# Copyright Notices

## C and C++ Libraries for Parsing PDB Records

# Glossary

**alignment**—The optimal matching of residue positions between sequences, typically a query sequence and one or more template sequences.

**anchor**—A constraint on alignment set at a given residue position. Alignment changes must preserve the query-template pairing at that residue until the anchor is removed.

**ASL**—Atom Specification Language.

**cofactor**—The complement of an enzyme reaction, usually a metal ion or metal complex.

**Comparative Modeling**—Protein structure modeling based on a query-template match with a substantial percentage of identical residues (usually 50% or greater sequence identity).

**composite template**—A type of template used in the Threading path, produced from the core (invariable) and variable regions of a family of structurally similar proteins.

**constraints**—Tools to keep regions of a sequence (alignment constraints) or structure (during minimization) in a particular configuration.

**deletions**—The residues missing from a query sequence that are present in a template sequence.

**Fold Recognition**—The use of secondary structure matching and profiles generated from multiple sequence/structure alignments to find templates when sequence methods are unsuccessful.

**gaps**—The spaces in an alignment resulting from insertions and deletions.

**HETATOMs**—The atoms of residues, including amino acids, that are not one of the standard 20 amino acids. In PDB files, HETATM.

**homolog**—A sequence/structure related to the query sequence; i.e., a sequence with many of the same residues in the same patterns as the query sequence. Usually these sequences are derived from the same family and may have similar function.

**insertions**—The extra residues found in a query sequence that are not found in a template sequence.

**ligand**—A small molecule that binds to a protein; e.g., an antigen, hormone, neurotransmitter, substrate, inhibitor, and so on.

**loop**—A region of undefined secondary structure.

**project**—A collection of related data, such as structures with their associated properties. In Prime a project comprises one or more *runs* (executions of the Prime workflow). The project may include data that does not appear in the *project table*.

**project table**—The Maestro panel associated with a project, featuring a table with rows of entries and columns of properties. In Prime, project table entries are usually model structures.

**query sequence**—A sequence of unknown structure or fold.

**Ranking Score**—The score used to rank composite templates derived from different seed templates. Generated by the Global Scoring Function in the Threading Path.

**refinement**—An improvement of a model structure through energy-based optimization of selected regions.

**run**—A single execution of the Prime workflow using a particular set of choices (of templates, of paths, and of settings). Each run belongs to a *project*. Runs cannot be saved without saving the project to which they belong.

**SSA**—Secondary structure assignment.

**SSP**—Secondary structure prediction.

**template sequence**—A sequence of known structure and fold used as a basis for building a model of the query.

**Threading**—A structure prediction process in which Fold Recognition is used to define templates, then backbone models are built via alignment to composite templates and refined. May be used when query-template sequence identity is low.

**Workspace**—The open area in the center of the Maestro window in which structures are displayed.

**Z-Score**—Measures the compatibility of the query sequence with the model structure, relative to the compatibility of randomly shuffled sequences of the same composition.

# Index